AMSTRAD CPC 464
«ARNOLD»

DEI
s.p.a.

AMSTRAD CPC 464
«ARNOLD»

MANUALE D'USO

s.p.a.

Largo Porta Nuova, 14 - 24100 BERGAMO - Tel. 035/221031 (5 linee r.a.)

INDICE

Manuale d'uso

Questa guida è divisa in tre sezioni principali: la prima specificatamente scritta per introdurre i neofiti nel mondo dei computer, per rendere ad essi familiari i principali concetti e la terminologia di base. Se non siete mai arrivati a scrivere un programma, anche breve, per un computer è meglio che iniziate la lettura di questo manuale dalle prime pagine. Chi invece ha già avuto precedenti esperienze può cominciare dal CAPITOLO 1.

IMPORTANTE

1. Per ogni connessione riferitevi sempre a quanto contenuto nella sezione intitolata "INSTALLAZIONE".
2. Non collegate mai al CPC 464 un elemento non compreso fra quelli citati in questo manuale. Potreste altrimenti danneggiare seriamente il computer e invalidare la garanzia.
3. Tenete il più lontano possibile sia dalla tastiera che dal monitor qualunque cosa contenga liquidi [vasi da fiori, caraffe, ...]
4. Non coprite mai le feritoie di ventilazione presenti nel CPC 464 e nel Monitor.
5. Assicuratevi sempre di aver registrato i vostri programmi prima di spegnere il CPC 464. Circa i metodi da seguire per il salvataggio dei programmi riferitevi al CAPITOLO 2, dopo aver terminato la lettura dei "FONDAMENTI".
6. Evitate di utilizzare cassette al Cromo o al Metal e di lunghezza superiore ai 60 minuti.
7. Prima di registrare i vostri programmi avvolgete il nastro oltre la 'coda', solitamente rossa.
8. Per assicurarvi il corretto funzionamento del CPC 464, in ogni sua parte, tenetelo sempre lontano da fonti di calore e da campi magnetici come quelli prodotti da altoparlanti e motori elettrici, soprattutto se di grandi dimensioni.
9. Qualunque danno accada al vostro CPC 464 non cercate mai di porvi rimedio da soli: rivolgetevi sempre al nostro Servizio di Assistenza.

FONDAMENTI 1

INSTALLAZIONE

Il personal computer AMSTRAD CPC 464 può essere installato utilizzando:

AMSTRAD GT64 / CTM640	Monitor a fosfori verdi/colore
AMSTRAD MP1	Modulatore / Adattatore di rete più un qualunque apparecchio TV.

Riferitevi alla Sezione appropriata per l'installazione del vostro sistema e procedete seguendo le istruzioni.

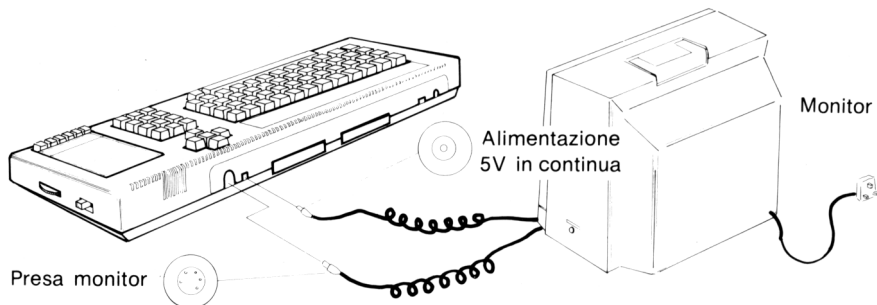



Figura 1

1.1 — AMSTRAD GT64/CTM640 Monitor a fosfori verdi/colore

Togliete dall'imballaggio il monitor prescelto e collegate una spina ai terminali del cavo dell'alimentazione.

Non è necessario alcun collegamento interno, non cercate quindi di aprire né il monitor né, tantomeno, il CPC 464.

Il computer deve essere posto di fronte al monitor, come mostrato in Figura 1. Il connettore più grosso — DIN a 6 poli — deve essere inserito nella presa marcata 'MONITOR' presente sul retro del CPC 464. Il secondo cavo che esce dal mo-

nitor, quello con lo spinotto più piccolo, deve essere inserito nell'ingresso segnato '5V DC', sempre sul retro del computer.

Assicuratevi che il pulsante di accensione del monitor sia in posizione di OFF e inserite la spina nella presa di rete più vicina al vostro tavolo. Ora accendete il monitor e portate in posizione di ON l'interruttore presente sul lato destro del CPC 464. Il led rosso presente sulla parte alta della tastiera si accenderà e sul monitor comparirà il messaggio:

```
Amstrad 64K Microcomputer  (v1)
©1984 Amstrad Consumer Electronics plc
          and Locomotive Software Ltd.
BASIC 1.0
```

Per evitare di stancare inutilmente la vostra vista utilizzate i controlli di luminosità del monitor. In effetti il monitor a fosfori verdi (GT64) è dotato di tre controlli: quello marcato 'Brightness' vi permette di regolare la luminosità, con quello marcato 'Contrast' regolate il contrasto e con l'ultimo, marcato 'V hold' potete centrare l'immagine sullo schermo. Il monitor a colori invece (CTM640) ha unicamente la possibilità di regolare la luminosità con il controllo posto sul fianco destro in basso.

1.2 — AMSTRAD MP1 modulatore/adattatore di rete

L'MP1 è un accessorio che permette di utilizzare il CPC 464 in connessione con il vostro TV Color domestico.

Estraete l'MP1 dall'imballaggio e collegate al cavo dell'alimentazione una spina.

Non sono necessarie connessioni interne, non dovete perciò cercare di aprire l'MP1 o il CPC 464.

Il modulatore/adattatore deve essere posto alla destra del computer, come mostrato in Figura 2. Dall'MP1 escono quattro cavi: quello con il connettore più grande — DIN a 6 poli — deve essere collegato alla presa marcata 'MONITOR'

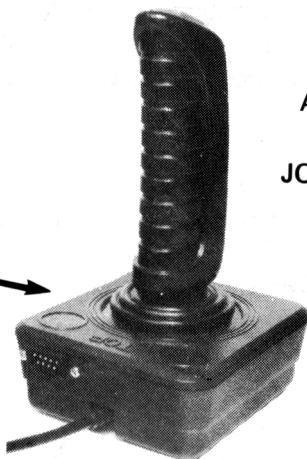
sul retro del computer, mentre quello con lo spinotto più piccolo va inserito nella presa marcata '5V DC', sempre sul retro del computer. Gli altri due cavi vanno uno alla presa di rete e l'altro all'ingresso d'antenna del televisore.

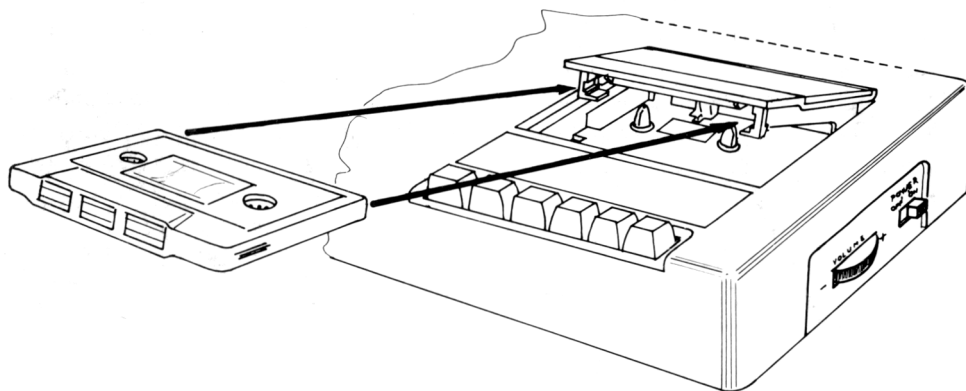
Assicuratevi che l'interruttore di accensione del computer sia in posizione 'OFF' e connettete l'MP1 ad una presa di corrente. Ora accendete l'apparecchio TV e il CPC 464, spostando l'apposito interruttore presente sulla destra del computer in posizione 'ON'.

All'accensione del led rosso presente nella parte alta della tastiera sintonizzate il TV intorno al canale 36 UHF, sino a quando sul video non vedrete il messaggio:

A questo punto eseguite la regolazione fine di sintonia, contrasto e luminosità sul televisore fino a quando il messaggio non apparirà in un bel colore giallo oro su sfondo blu intenso.

1.3 — Joystick

Il joystick AMSOFT, modello JY1, è un accessorio che può risultare utile nei giochi di animazione, ma solo in quelli predisposti per accettare tale tipo di controllo. Il cavo del joystick ha un connettore particolare per poter essere inserito nella presa chiamata 'USER PORTS (I/0)'. Il CPC 464 può utilizzare sino a due joystick, il secondo dei quali dovrà essere connesso al primo tramite il connettore presente nella base di ogni joystick JY1.

1.4 — Cassetta di presentazione

Nella confezione del vostro Amstrad CPC 464 è compresa una cassetta di presentazione. Aprite lo sportello del Datacorder premendo il tasto marcato [STOP/EJECT], poi inserite la cassetta nel Datacorder come mostrato in Figura 3 e assicuratevi che la scritta 'SIDE 1' sia rivolta verso l'alto:

Chiudete lo sportello del Datacorder e premete il tasto [REW] per assicurarvi che il nastro sia riavvolto sino all'inizio. Appena il nastro si ferma premete il tasto [STOP/EJECT] e azzerate il contanastro con il bottone marcato [COUNTER RESET]. Ora, premete contemporaneamente il tasto marcato [CTRL] (Control) e il piccolo [ENTER] che si trova in basso a destra nel tastierino numerico accanto al Datacorder. Sullo schermo apparirà il messaggio:

```
RUN"
Press PLAY then any key :
```

Avviate il Datacorder premendo il tasto PLAY sino a quando non rimane fisso in posizione abbassata e di seguito premete un tasto qualsiasi.
Il nastro inizierà a girare e dopo poco vedrete questo messaggio apparire sullo schermo:

Loading WELCOME 1 block 1

Occorreranno circa 5 minuti per leggere l'intero nastro. Con il passare del tempo il numero che segue la parola 'block' cambierà in 2, 3, ecc., sino a quando il nastro non si fermerà. A questo punto inizierà automaticamente l'esecuzione del programma di presentazione. Sedetevi e state a guardare. Una volta terminata la prima esecuzione il programma ripartirà automaticamente: per interromperlo dovete premere due volte consecutivamente il tasto [ESC]. Estraete la cassetta dal Datacorder premendo il tasto [STOP/EJECT] e giratela per caricare la seconda facciata ripetendo esattamente la procedura seguita per la prima.

1.5 — Caricamento di altre cassette

La CASSETTA DI PRESENTAZIONE può essere caricata solo seguendo il procedimento descritto nel paragrafo precedente. I programmi BASIC non protetti possono essere invece caricati in diversi modi. Riavvolgete sempre il nastro sino all'inizio e appena si ferma premete il tasto [STOP/EJECT].
Per resettare totalmente il computer e pulire la memoria è necessario premere i tasti [CTRL], [SHIFT] e [ESC] nell'ordine, tenendo premuti i primi due sino a quando non avrete schiacciato anche il tasto [ESC]. Lo schermo si pulirà ed apparirà il messaggio iniziale, come se aveste appena acceso il computer.
Nelle istruzioni che seguono, la dicitura [ENTER] indica che dovete premere uno dei due tasti marcati 'ENTER' e non scrivere la parola ENTER ! Il simbolo " — apici — si ottiene premendo lo [SHIFT] e il tasto '2' nella fila superiore della tastiera.

Scrivete:

load " " [ENTER]

Il computer risponde...

Press PLAY then any key:

Ora premete il tasto PLAY del Datacorder e poi un qualunque tasto.
Il nastro inizierà a girare e dopo poco tempo vedrete apparire sullo schermo il messaggio:

Loading <nome programma> block 1

Il numero di blocco continuerà ad incrementarsi fino al termine del caricamento, quando il messaggio:

Ready

apparirà sullo schermo.

Alternativamente, potete specificare il nome del programma che intendete caricare. Per fare questo, scrivete:

load " <titolo>" [ENTER]

Il computer risponde

Press PLAY then any key:

Premete il tasto [PLAY] del Datacorder e poi un tasto qualsiasi.
Il nastro inizierà a muoversi. Se il programma che avete chiesto di caricare non si trova all'inizio, il computer lo cercherà sul nastro. Ponete attenzione nello scrivere il nome del programma correttamente.

Se durante la ricerca del titolo da voi richiesto il computer incontra titoli differenti, vedrete comparire sullo schermo il messaggio:

Found <altro titolo> block 1

Il computer non caricherà questo programma, ma continuerà a ricercare il titolo da voi specificato. Per forzare l'interruzione della ricerca potete premere il tasto [ESC].

Una volta trovato il titolo da voi richiesto apparirà il messaggio:

Loading <titolo> block 1

Il numero di blocco verrà incrementato durante il caricamento, al termine del quale sullo schermo apparirà la scritta:

Ready

Ora scrivete:

run [ENTER]

in questo modo il programma appena caricato verrà eseguito.
Per eseguire un programma direttamente, senza dover prima specificare la richiesta di caricamento, scrivete semplicemente:

run " " [ENTER]

e il computer risponde con il solito:

Press PLAY then any key:

Dopo aver premuto il [PLAY] seguito da un qualsiasi tasto il computer cercherà il programma. Trovatolo, lo caricherà e lo eseguirà senza attendere nuove istruzioni da tastiera. Potete interrompere la sequenza premendo il tasto [ESC], come già detto.

1.6 — Caricamento di programmi pre - registrati

Le istruzioni date sin qui vi permetteranno di caricare la maggioranza dei programmi disponibili per il CPC 464.

Comunque, riferitevi sempre alle istruzioni specifiche incluse in ogni cassetta pre - registrata.

1.7 — Salvataggio di programmi su cassetta

Un programma può essere salvato (registrato) su cassetta per essere poi riutilizzato. Inserite nel Datacorder una cassetta da cui non siano state tolte le placche di protezione presenti sul retro e scrivete:

save "<titolo>" [ENTER]

Il computer risponderà con:

Press REC and PLAY then any key:

Premete i tasti [REC] e [PLAY] del Datacorder sino a quando questi non rimangono in posizione abbassata, poi un tasto qualunque.

Il computer risponderà con:

Saving <titolo> block 1

Una volta terminato il salvataggio del programma vedrete apparire sullo schermo il messaggio:

Ready

Fermate il registratore premendo il tasto [STOP/EJECT].

I programmi commerciali vengono normalmente registrati su cassetta con l'opzione di protezione. In questo caso non è possibile salvare i programmi su di una propria cassetta.

FONDAMENTI 2
LA TASTIERA

Illustreremo la funzione di alcuni tasti del computer. Chi ha già esperienza può passare direttamente alla Sezione successiva.

[**ENTER**]

Ci sono due tasti [ENTER]. Ognuno di essi trasmette al computer tutto ciò che avete scritto. Dopo la pressione del tasto [ENTER], il computer va automaticamente a capo. Ogni linea di istruzione data al computer deve essere seguita da un [ENTER]. D'ora in avanti, la pressione del tasto ENTER dopo ogni istruzione o riga di programma sarà indicata con [ENTER].

[**DEL**]

Questo tasto è utilizzato per cancellare il carattere che si trova sullo schermo alla sinistra del cursore.

Scrivete ' a b c d ' e vedrete che la lettera d si trova alla sinistra del cursore. Per cancellarla premete [DEL] una volta sola. Tenendo premuto più a lungo il tasto [DEL] cancellerete anche le lettere a b c.

[**SHIFT**]

Ci sono due tasti [SHIFT]. Premendo uno di questi due assieme ad un altro tasto vedrete comparire sullo schermo una lettera maiuscola oppure il simbolo riportato nella parte superiore dei tasti a doppia marcatura.

Fate qualche prova: battendo [SHIFT] e 'a' vedrete comparire una lettera 'A', battendo [SHIFT] e '2' vedrete comparire i doppi apici.

[**CAPS LOCK**]

Questo tasto opera in un modo simile allo [SHIFT] e ha la funzione equivalente a quella del tasto 'Fissa maiuscole' delle macchine da scrivere. È sufficiente premerlo una sola vclta perché tutte le lettere battute risultino scritte in maiuscolo. Questa funzione non ha effetto sui tasti numerici in quanto i simboli riportati sulla parte superiore di detti tasti non verranno rappresentati. Premete [CAPS LOCK] una sola volta e scrivete:

abcdef123456

Noterete sullo schermo che tutte le lettere sono maiuscole, mentre i numeri non sono divenuti simboli, come avviene con lo [SHIFT]. Volendo ottenere la rappresentazione dei simboli è necessario utilizzare lo [SHIFT] anche quando il [CAPS LOCK] è operativo. Premete i tasti seguenti assieme allo [SHIFT]:

ABCDEF123456

Sullo schermo vedrete:

ABCDEF!"#$%&

Volendo tornare alle lettere minuscole è sufficiente premere nuovamente [CAPS LOCK] una sola volta.

Se volete ottenere le lettere maiuscole e i simboli della fila superiore senza dover continuamente utilizzare lo [SHIFT], è sufficiente premere i tasti [CTRL] e [CAPS LOCK]. Ora scrivete:

abcdef123456

Sullo schermo vedrete:

ABCDEF!#$%&

Per ottenere la rappresentazione dei numeri nella situazione appena descritta utilizzate il tastierino numerico posto alla destra della tastiera principale.

Premendo nuovamente i tasti [CTRL] e [CAPS LOCK] tornerete al modo in cui vi trovavate precedentemente.

[**CLR**]

Questo tasto è utilizzato per cancellare il carattere che si trova alla posizione del cursore.

Scrivete ABCDEFGH. Il cursore si posizionerà alla destra dell'ultimo carattere battuto (H). Ora premete il tasto marcato [←] quattro volte. Il cursore si muoverà di quattro posizioni verso sinistra, sovrapponendosi alla lettera E. Notate come la

lettera E sia visibile sotto il cursore. Premete [CLR] una volta e la lettera E verrà cancellata, mentre F, G e H si sposteranno di una posizione verso sinistra. Continuando a tenere premuto [CLR] cancellerete anche le lettere F, G e H.

[**ESC**]

Questo tasto è utilizzato per interrompere l'esecuzione di una funzione durante il suo svolgimento. Premendo una prima volta [ESC] si sospende momentaneamente il lavoro del computer, che potrà essere ripreso con la pressione di qualsiasi tasto diverso da [ESC].

Premendo [ESC] due volte si interrompe del tutto l'esecuzione del programma e si torna al modo diretto con il computer pronto ad accettare nuovi comandi.

IMPORTANTE

Quando raggiungete il margine destro dello schermo, il cursore si posiziona automaticamente all'inizio della riga seguente da cui riprende la visualizzazione dei caratteri. In altre parole, non dovete premere [ENTER] per andare a capo, nè qualcosa di equivalente al tasto 'ritorno carrello' delle macchine da scrivere.

Syntax Error

Con il messaggio 'Syntax error' il computer intende comunicarvi che non riesce ad interpretare l'istruzione inserita. Se, ad esempio, scriveste:

```
printt [ENTER]
```

otterreste in risposta il messaggio:

```
Syntax error
```

a significare che il computer non ha capito l'istruzione printt.

Se inserite un errore di battitura in una riga di programma, come in questo caso:

```
10 printt "abc" [ENTER]
```

l'errore di sintassi vi verrà segnalato solo quando l'istruzione verrà interpretata dal computer durante l'esecuzione del programma. Battendo infatti:

```
run [ENTER]
```

sullo schermo vedrete:

```
Syntax error in 10
10 PRINTT "abc"
```

Questo messaggio vi facilita la correzione dell'errore, presentandovi immediatamente la riga errata con il cursore posizionato all'inizio della stessa.

Per correggere, spostate il cursore verso destra sino a quando questo non si trova sopra la 't' non voluta e battete [CLR]. Alla pressione dell'[ENTER] la riga corretta verrà inserita nel programma. Per averne la prova date RUN e vedrete apparire sullo schermo la scritta 'abc'.

Introduzione ai comandi del BASIC AMSTRAD

Nel Capitolo 8 troverete una descrizione dettagliata di tutte le istruzioni dell'AMSTRAD BASIC; in questa Sezione ci limiteremo ad illustrare le più comuni.

CLS

Battete 'cls', sia in lettere minuscole che maiuscole, seguito da [ENTER]. Vedrete lo schermo pulirsi e la parola Ready apparire nell'angolo in alto a sinistra seguita dal cursore.

PRINT

È utilizzata per visualizzare caratteri, numeri e cifre sullo schermo. Inserite la seguente linea:

```
print "salve" [ENTER]
```

E sullo schermo vedrete:

```
salve
```

Gli apici " " racchiudono tutto ciò che il computer deve stampare. Scrivete cls [ENTER] per pulire lo schermo.

RUN

L'esempio precedente era composto di una sola riga, ma la maggior parte dei programmi richiede più righe, ognuna identificata da un numero che precede le istruzioni. Questi numeri indicano al computer l'ordine di esecuzione delle istruzioni. Scrivete:

```
10 print "salve" [ENTER]
```

In questo caso la scritta salve non viene stampata dopo la pressione del tasto [ENTER], perchè si tratta di una riga di programma e non di un comando diretto.

È quindi necessario battere:

run [ENTER]

per vedere 'salve' sullo schermo. L'istruzione print può essere abbreviata con ?, infatti:

10 ? "salve" [ENTER]
run [ENTER]

non provoca alcun errore.

LIST

Questo comando serve per avere sullo schermo la lista del programma presente in memoria. Infatti, battendo LIST [ENTER] comparirà l'unica riga di programma inserita, e cioè:

10 PRINT "salve"

Notate che ora la parola PRINT compare in lettere maiuscole, dato che il computer l'ha riconosciuta e accettata come comando del BASIC. Pulite nuovamente lo schermo con il CLS e ricordate che, così facendo, non provocherete la cancellazione della memoria.

GOTO

Tale comando permette di alterare la sequenza di esecuzione delle istruzioni, in modo da saltare un certo numero di righe o provocare una iterazione. Provate con:

20 GOTO 10 [ENTER]
run [ENTER]

e la parola 'salve' verrà visualizzata sino all'interruzione del programma tramite la duplice pressione del tasto [ESC].

Se volete ottenere la stampa continuata della parola 'salve' su tutte le linee dello schermo, fino al riempimento completo dello stesso, è sufficiente che chiudiate la riga 10 con un punto e virgola dopo gli apici. In questo modo avete detto al computer di stampare la parola 'salve' senza andare a capo ogni volta. Usando la virgola in luogo del punto e virgola, direste al computer di visualizzare la parola 'salve' ogni volta all'inizio della successiva zona di stampa. Le zone di stampa hanno un'ampiezza di default di 13 caratteri, pertanto dovete porre attenzione quando volete stampare stringhe più lunghe di 12 caratteri.

Per uscire dal programma battete [ESC] due volte consecutivamente, come al solito. Per cancellare la memoria battete in sequenza [SHIFT][CTRL][ESC].

INPUT

Questo comando è utilizzato per comunicare al computer di rimanere in attesa dell'inserimento di un dato: ad esempio, la risposta ad una domanda. Provate con:

```
10 input "quanti anni hai";anni [ENTER]
20 print "sembri più giovane di";anni"anni." [ENTER]
run [ENTER]
```

Sullo schermo apparirà:

Quanti anni hai?

rispondendo, ad esempio 18, il commento visualizzato sarebbe:

sembri più giovane di 18 anni.

L'esempio mostra l'uso del comando INPUT con una variabile numerica. Il numero fornito in risposta viene associato alla 'parola' anni e stampato secondo quanto specificato alla riga 20. Notate che il nome della variabile, anni in questo caso, non ha alcuna rilevanza e potrebbe essere anche una semplice lettera.

Se la risposta ad un input non è forzatamente numerica, ma può essere composta da qualsiasi combinazione di lettere, numeri e simboli, è obbligatorio aggiungere il simbolo $ alla fine della variabile. Battete il programma seguente:

```
10 input "Come di chiami";nome$ [ENTER]
20 print "ciao";nome$" il mio nome è Arnold" [ENTER]
run [ENTER]
```

sullo schermo vedrete:

Come ti chiami?

battete il vostro nome, ad esempio Roberto, e otterrete in risposta:

ciao Roberto il mio nome è Arnold

Per inciso, Arnold è stato il soprannome dato all'AMSTRAD CPC 464 dai progettisti.

Di nuovo notate che nome$ potrebbe essere sostituito ad esempio con a$.

Proviamo ora a unire i due esempi precedeznti in un solo programma:

```
5 cls          [ENTER]
10 input "Come ti chiami";a$ [ENTER]
20 input "Quanti anni hai";b [ENTER]
30 print "Devo dire ";a$" che non dimostri";
   b"anni"
run [ENTER]
```

Questo programma usa due variabili, a$ per il nome e b per l'età. Sullo schermo vedrete:

Come ti chiami?

battete il nome in risposta e, dopo la pressione dell'[ENTER], vedrete comparire la seconda domanda:

Quanti anni hai?

se avete risposto, ad esempio, Roberto e 18, il computer replicherà con:

Devo dire Roberto che non dimostri 18 anni

Editing di un programma

Nel caso vengano commessi errori nella battitura di un programma, ottenendo messaggi del tipo Syntax error o altri, potete correggere la riga invece di ribatterla dall'inizio. Provate, per esempio, a ribattere il programma precedente introducendo degli errori:

```
5 clss
10 input "come ti chiami"; a$ [ENTER]
20 input "quanti anni ai"; b [ENTER]
30 print "Devo dire"; a$" che non dimostri";
b "anni" [ENTER]
```

Abbiamo commesso tre errori:

alla riga 5 clss invece di cls
alla riga 20 ai invece di hai
alla riga 30 è stato omesso lo spazio tra la parola dire e i doppi apici.

Esistono tre metodi per eseguire l'editing di un programma. Il primo consiste semplicemente nel riscrivere la nuova riga con lo stesso numero di quella errata in modo da ottenerne la sostituzione. Di seguito vengono illustrati gli altri due metodi.

Funzione EDIT

Per correggere l'errore alla riga 5 battete:

edit 5 [ENTER]

otterrete la visualizzazione della riga 5 con il cursore posizionato sul numero di riga. Per togliere la seconda s da clss premete il tasto [→] fino a portare il curso-

re sopra di essa e premete di seguito il tasto [CLR]. A questo punto la seconda s è scomparsa e premendo [ENTER] memorizzerete la riga corretta. Per avere la controprova battete:

list [ENTER]

Funzione COPY CURSOR

Per correggere gli errori alle righe 20 e 30 tenete premuto il tasto [SHIFT] e usate il tasto [↑] per posizionare il cursore di COPY all'inizio della riga 20. Notate come il cursore principale non si sia spostato e come compaiano a questo punto due cursori sullo schermo. Ora premete il tasto [COPY] fino a portare il cursore di COPY sulla a della parola 'ai'. Notate che la riga 20 viene riscritta sotto l'ultima riga del programma e che il cursore principale si trova alla stessa posizione del cursore di COPY. Adesso potete battere la h che manca, che comparirà solo sulla prima delle due righe di numero 20.

Il cursore principale si è spostato, mentre il cursore di COPY è rimasto fermo. Proseguite con il tasto [COPY] fino ad ottenere la riscrittura completa della riga 20 e premete [ENTER] per memorizzare la riga corretta. Il cursore di COPY scompare e il cursore principale si posiziona sotto la nuova riga 20. Per correggere il secondo errore, tenete premuto [SHIFT] e portate il cursore di COPY con il tasto [↑] all'inizio della riga 30. Con il tasto [↑COPY] posizionate il cursore di COPY sui doppi apici che si trovano a destra della parola 'dire' e premete di seguito la barra spaziatrice una sola volta. Avete ottenuto l'inserimento di uno spazio nella seconda delle due righe di numero 30. Ora, sempre con il tasto [COPY], riscrivete completamente la riga 30 e premete [ENTER]. Potete controllare l'avvenuta correzione del programma con il comando: list [ENTER].

Ora resettate il computer con la sequenza [CTRL] [SHIFT] e [ESC].

IF THEN

Ora ampliamo il programma precedente con l'utilizzo dei comandi IF e THEN.

Nel programma che segue si trovano due simboli a voi nuovi: <, che significa 'minore di' ed è posto a fianco del tasto M e il simbolo >, che significa 'maggiore di' e si trova accanto al precedente. Inserite il programma:

```
5 cls          [ENTER]
10 input "Come ti chiami"; a$ [ENTER]
20 input "Quanti anni hai"; età [ENTER]
30 if età < 13 then 60 [ENTER]
40 if età < 20 then 70 [ENTER]
50 if età > 19 then 80 [ENTER]
```

```
60 print "Caro" a$ "sei ancora un ragazzino a" età
"anni": end [ENTER]
70 print "Caro" a$ "sei un giovanotto a" età
"anni": end [ENTER]
80 print "Caro" a$ "cominci ad invecchiare a" età
"anni": end [ENTER]
```

Verificate che il programma sia stato inserito correttamente listandolo, date il run e rispondete alle domande che il computer vi pone. In questo modo potrete rendervi conto di come agiscono la IF e la THEN in un programma. Notate che le righe 60 e 70 terminano con il comando END, che indica letteralmente la fine dell'elaborazione. Omettendo questo comando alla riga 60, il programma eseguirebbe anche le righe 70 e 80.

Come potete rilevare, per separare due comandi inseriti sulla stessa riga deve essere utilizzato il simbolo dei due punti [:]. La riga 5 serve per pulire lo schermo all'inizio del programma e vi consigliamo di seguire questa prassi come abitudine. Altri comandi associati alla IF e alla THEN sono ELSE e GOTO, il cui uso diverrà chiaro proseguendo nella lettura di questo manuale.

FOR TO NEXT

Ecco un esempio di come un computer può stampare la tabellina del 12. Fate solo attenzione ad usare il simbolo * per indicare la moltiplicazione:

```
5 cls [ENTER]
10 for a=1 to 20 [ENTER]
20 print a "* 12 =" a*12 [ENTER]
30 next a
run [ENTER]
```

Per mettere un po' d'ordine nell'incolonnamento, apportate le seguenti modifiche:

```
10 for a=1 to 9 [ENTER]
40 for a=10 to 20 [ENTER]
50 print a"* 12 =" a*12 [ENTER]
60 next a
```

Utilizzate il programma con altri numeri. Ad esempio per ottenere la tabellina del 17, basta sostituire alle righe 20 e 50 il numero 12 con il 17. Quando avete finito date il solito reset con la sequenza [CTRL] [SHIFT] [ESC].

Tenete presente che è possibile specificare l'incremento della variabile, in questo caso a, utilizzando il comando STEP, per il quale rimandiamo al Capitolo 8.

Un po' di aritmetica

Nulla di più facile che utilizzare il CPC 464 come una calcolatrice. Gli esempi che seguono servono a dimostrarlo:

Addizione (il simbolo + si ottiene con [SHIFT] e ;)

```
?3+3 [ENTER]
6
```

notate che non è necessario battere il simbolo =

Sottrazione (il simbolo – si trova sullo stesso tasto di =)

```
?7-4 [ENTER]
3
```

Moltiplicazione (il simbolo * si ottiene con [SHIFT] e :)

```
?3*4 [ENTER]
12
```

Divisione (il simbolo / si trova sullo stesso tasto di ?)

```
?20/5 [ENTER]
 4
```

Radice quadrata

```
?sqr (16) [ENTER]
 4
```

ricordate di porre il numero di cui volete estrarre la radice quadrata tra parentesi rotonde.

Elevamento a potenza (utilizzate il tasto ↑)

```
?3↑3 [ENTER]
 27
```

Radice cubica

Per calcolare la radice cubica di un numero è sufficiente elevarlo alla potenza 1/3. Ad esempio, per ottenere la radice cubica di 27 battete:

```
?27 ↑ (1/3) [ENTER]
 3
```

Espressioni complesse

Le espressioni vengono calcolate secondo la priorità degli operatori: elevamento a potenza, moltiplicazione, divisione, addizione, sottrazione. Espressioni più complesse vengono esaminate nel seguito.

Se volete calcolare:

```
3+7-2*7/4
```

il comportamento del computer sarà:

```
3+7-2*7/4
=3+7-14/4
=3+7-3.5
=10-3.5
=6.5
```

Infatti:

```
?3+7-2*7/4 [ENTER]
 6.5
```

La priorità tra gli operatori può essere modificata tramite l'uso di parentesi che vengono sempre risolte prima di ogni altro calcolo. Provate con:

```
?(3+7-2)*7/4 [ENTER]
 14
```

Ancora sull'elevamento a potenza

Se i vostri calcoli prevedono numeri molto grandi o molto piccoli, può essere utile la notazione scientifica, in cui la lettera E indica l'elevamento a potenze di 10. Ad esempio, dato che 300 equivale a 3×10↑2, in notazione scientifica scriveremmo: 3E2 oppure 3e2. Allo stesso modo 0.03 diverebbe 3e-2. Seguono alcuni esempi:

```
?30*10        [ENTER]     ?3E1*1E1 [ENTER]
 300                       300

?3000*1000 [ENTER]        ?3E3*1E3 [ENTER]
 3000000                   3000000

?3000*0.001 [ENTER]       ?3E3*1E-3 [(ENTER]
 3                         3
```

FONDAMENTI 3

GRAFICA E SUONO

Il personal computer AMSTRAD CPC 464 è in grado di operare secondo tre 'modi di schermo', rispettivamente identificati con Modo 0, Modo 1 e Modo 2. All'accensione viene automaticamente selezionato il Modo 1.

Per comprendere le differenze fra i tre modi di schermo, accendete il computer e tenete premuto il tasto del numero 1 fino a riempire completamente due linee. Provate a contare quanti numeri 1 sono contenuti in ogni linea dello schermo e vedrete che sono 40, a dimostrazione del fatto che il Modo 1 vi mette a disposizione 40 colonne per riga. Premendo il tasto [ENTER] otterrete ora il messaggio Syntax error, ma non preoccupatevi: abbiamo semplicemente utilizzato un metodo veloce per riportare il computer in stato di Ready, in attesa dell'istruzione successiva.

A questo punto battete: mode 0 [ENTER].

Notate come i caratteri rappresentati sullo schermo abbiano dimensioni maggiori rispetto a prima. Se riempite nuovamente due righe con il numero 1 e rifate il conto, arriverete facilmente alla conclusione che il Modo 0 vi mette a disposizione 20 colonne per riga.

Date di nuovo [ENTER] e battete: mode 2 [ENTER].

La dimensione dei caratteri è ora la minima possibile, mentre il numero di colonne per riga è pari a 80. Ricapitolando:

Modo 0 = 20 colonne
Modo 1 = 40 colonne
Modo 2 = 80 colonne

COLORI

I colori disponibili sono 27. Nel caso utilizziate il monitor a fosfori verdi GT 64, i colori sono rappresentati come diverse gradazioni di verde. Ricordate che potete abbinare al GT 64 il Modulatore/Adattatore di rete MP1 e sfruttare il vostro TV color per il colore.

In Modo 0, potete mandare sullo schermo contemporaneamente 16 dei 27 colori disponibili.

In Modo 1, potete mandare sullo schermo contemporaneamente 4 dei 27 colori disponibili.

In Modo 2, potete mandare sullo schermo contemporaneamente 2 dei 27 colori disponibili.

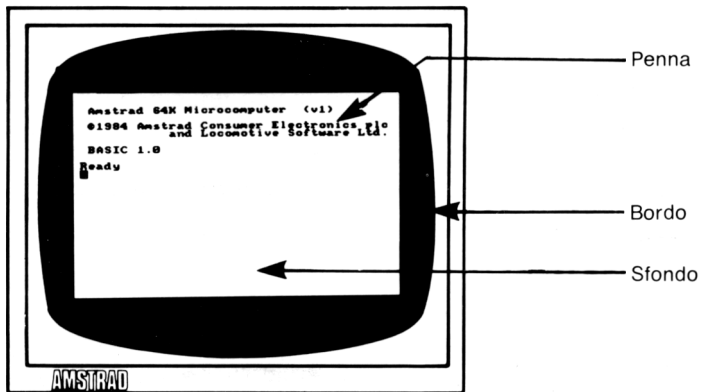
I colori del bordo (BORDER), dello sfondo (PAPER) e della penna (PEN) possono essere cambiati a vostro piacimento, in maniera indipendente l'uno dall'altro.

La tabella che segue elenca i 27 colori disponibili, ciascuno con il proprio numero di INK.

TAVOLA DEI COLORI

Numero Ink	Colore/Ink	Numero Ink	Colore/Ink
0	Nero	14	Blu Pastello
1	Blu	15	Arancio
2	Blu Chiaro	16	Rosa
3	Rosso	17	Magenta Pastello
4	Magenta	18	Verde Chiaro
5	Malva	19	Verde Mare
6	Rosso Chiaro	20	Ciano Chiaro
7	Porpora	21	Verde Lime
8	Magenta Chiaro	22	Verde Pastello
9	Verde	23	Ciano Pastello
10	Ciano	24	Giallo Chiaro
11	Blu Cielo	25	Giallo Pastello
12	Giallo	26	Bianco Chiaro
13	Bianco		

All'accensione i colori del bordo (BORDER), dello sfondo (PAPER) e della penna (PEN) sono così selezionati:

Bordo	Colore N. 1 (blu)
Sfondo	Colore N. 1 (blu)
Penna	Colore N. 24 (giallo chiaro)

La figura precedente mostra chiaramente che i caratteri vengono scritti nel colore della PEN su quella parte di schermo definita PAPER e circondata dal BORDER. Ricordate che all'accensione i colori del bordo e dello sfondo coincidono.

In altre parole, potete associare i concetti di PEN e PAPER sullo schermo ad una penna stilografica e ad un foglio di carta. Così come il colore in cui scrive la penna stilografica sul foglio di carta è determinato dal colore dell'inchiostro con cui avete riempito il serbatoio, allo stesso modo il colore dell'INK della PEN può essere cambiato e definisce il colore dei caratteri sullo schermo. Anche il foglio di carta può essere di diversi colori, proprio come può variare il colore del PAPER sullo schermo.

Per variare il colore del BORDER usate il comando:

border 0 [ENTER]

e vedrete il bordo passare dal blu al nero, dato che lo 0 corrisponde al nero, come mostrato dalla tabella. Divertitevi con gli altri colori e poi battete:

cls [ENTER]

per pulire lo schermo.

Il procedimento per cambiare i colori di PAPER e PEN è simile a quello già mostrato. Provate con:

paper 2 [ENTER]
cls [ENTER]
pen 3 [ENTER]

Per comprendere tutti i dettagli di quanto è successo, riferitevi anche alla Tabella 2 che segue. All'accensione il computer seleziona automaticamente lo 0 come codice dell'INK del PAPER. Detto codice, come risulta appunto dalla Tabella 2, corrisponde al colore N. 1, che è il blu. Questo significa che il colore del PAPER all'accensione è il blu. La stessa Tabella 2 mostra come ciò sia valido, nel caso specifico, per tutti e tre i Modi di schermo.

In precedenza abbiamo assegnato al PAPER l'INK di codice 2. In questo caso la Tabella 2 vi dice che in Modo 1 il codice 2 corrisponde al colore numero 20, cioè il ciano chiaro.

COLORI DEGLI INK

PAPER/PEN	Modo 0	Modo 1	Modo 2
0	1	1	1
1	24	24	24
2	20	20	1
3	6	6	24
4	26	1	1
5	0	24	24
6	2	20	1
7	8	6	24
8	10	1	1
9	12	24	24
10	14	20	1
11	16	6	24
12	18	1	1
13	22	24	24
14	lampeggio 1,24	20	1
15	lampeggio 16,11	6	24

La PEN usata all'accensione è la numero 1, che corrisponde in Modo 1 al colore numero 24, cioè il giallo chiaro. Questo è il colore dei caratteri sullo schermo all'accensione del computer. Cambiando ad esempio il numero della PEN da 1 a 3, otterrete, in Modo 0, i caratteri in colore 6, cioè rosso chiaro.

I colori del PAPER e della PEN possono essere cambiati con il comando INK, nella forma INK n,m dove n indica il numero della PEN o del PAPER a cui si vuole cambiare colore e m definisce il colore stesso. Ad esempio proviamo a cambiare il colore della PAPER 2 in nero e il colore della PEN 3 in bianco chiaro. Dalla Tabella 1 si ricava che il nero corrisponde a 0 e il bianco chiaro a 26. Pertanto:

ink 2,0 [ENTER]

(ricordate che stiamo usando il PAPER 2)

ink 3,26 [ENTER]

(ricordate che stiamo usando la PEN 3).

Ora, resettando il computer con la sequenza [CTRL] [SHIFT] e [ESC], riporterete i valori di INK, PEN e PAPER a quelli dell'accensione. Per cambiare immediatamente il colore dei caratteri (PEN) e dello sfondo (PAPER) battete, ad esempio:

ink 0,0 [ENTER]
ink 1,26 [ENTER]

e otterrete caratteri bianchi su fondo nero.

Lampeggio

È possibile far lampeggiare i caratteri tra due colori aggiungendo un secondo numero di colore al comando INK per la PEN. Per ottenere il lampeggio tra il bianco chiaro ed il rosso chiaro battete:

ink 1,26,6 [ENTER]

In questo caso 1 è il numero della PEN, 26 è il colore bianco chiaro, mentre 6 è il colore rosso chiaro. Anche il colore di fondo può lampeggiare, usando la stessa tecnica di aggiungere un secondo numero di colore al comando INK per il PAPER. Battendo, ad esempio:

ink 0,9,24 [ENTER]

vedrete lo sfondo lampeggiare tra il verde ed il giallo chiaro, dato che 0 è il numero di PAPER, 9 rappresenta il verde e 24 il giallo chiaro.

Notate che in Modo 0 gli INK di codice 14 e 15 sono automaticamente definiti come lampeggianti e quindi non richiedono di specificare un secondo numero di colore nel comando INK. Provate con:

mode 0 [ENTER]
pen 15 [ENTER]
paper 14 [ENTER]

Naturalmente questo lampeggio automatico avviene tra due colori predefiniti, ma avete sempre la possibilità di variarli. Ad esempio:

ink 15,0,26 [ENTER]

Infine facciamo lampeggiare anche il bordo:

border 6,9 [ENTER]

per ottenere il lampeggio tra il rosso chiaro e il verde.

Resettate il computer con la solita sequenza [CTRL] [SHIFT] [ESC].

PROGRAMMA DIMOSTRATIVO

Il programma che segue è una dimostrazione delle possibilità cromatiche del CPC 464. Abbiamo aggiunto anche alcuni effetti sonori per i quali si rimanda al seguito del manuale.

```
 10 MODE 0: INK 0,2:INK 1,24: PAPER 0
 20 PEN 1:FOR b=0 TO 26:BORDER b
 30 LOCATE 3,12:PRINT "COLORE DEL BORDO";b
 40 SOUND 4,(40-b)
 50 FOR t=1 TO 600:NEXT t:NEXT b:CLS
 60 FOR p=0 TO 15:PAPER p:PEN 5:PRINT "sfondo"; p:PRINT
 70 FOR n=0 TO 15:PEN n:PRINT "colore dei caratteri"; n
 80 SOUND 1,(n*20+p)
 90 FOR t=1 TO 100:NEXT t:NEXT n
100 FOR t=1 TO 1000:NEXT t:CLS:NEXT p
110 CLS:PAPER 0:PEN 1:LOCATE 7,12:PRINT "F I N E":FOR t=1 TO
    2000:NEXT t
120 MODE 1:BORDER 1:INK 0,1:INK 1,24:PAPER 0:PEN 1
```

Ricordatevi di battere [ENTER] alla fine di ogni riga.

GRAFICA

Da questo punto in avanti non useremo più mettere [ENTER] alla fine di ogni riga di programma, dato che dovrebbe ormai essere automatico.

Il set di caratteri del CPC 464 può essere visualizzato con il breve programma che segue:

```
10 for n=32 to 255: print n;chr$(n);
20 next
run
```

Alcuni dei caratteri sono simboli grafici, come ad esempio il numero 250 che rappresenta un omino che cammina verso destra. Per i dettagli consultate l'Appendice III alla fine di questo manuale.

LOCATE

Questo comando viene utilizzato per posizionare il cursore sullo schermo, tramite la definizione delle coordinate di riga e colonna. Normalmente il cursore si trova all'inizio sull'angolo in alto a sinistra dello schermo, che corrisponde alle coordinate x=1 e y=1 (x è la posizione orizzontale e y quella verticale). Ricordando che in Modo 1 disponete di 40 colonne e 25 righe, per visualizzare un carattere al centro della prima linea dovremo indicare come coordinate 20 per la x e 1 per la y:

mode 1 .... lo schermo si pulisce e il cursore si posiziona in alto a sinistra

```
10 locate 20,1
20 print chr$(250)
run
```

Giusto per avere una riprova che l'omino si trova sulla prima linea, battete border 0: il bordo diventerà nero permettendovi la verifica visiva.

In Modo 0, mentre il numero delle linee rimane pari a 25, il numero delle colonne disponibili si riduce a 20. Se battete:

```
mode 0
run
```

vedrete che l'omino si trova sempre sulla prima linea, ma non più nel centro, bensì all'estrema destra della linea, dato che la coordinata x=20 è quella dell'ultima colonna disponibile in Modo 0.

In Modo 2, dove le colonne diventano 80, mentre il numero delle linee non varia, lo stesso programma farà apparire l'omino in posizione ancora diversa. Provate a indovinare e se non ci riuscite battete pure:

```
mode 2
run
```

tornate ora al Modo 1 con:

```
mode 1
```

Esercitatevi da soli con il comando LOCATE e utilizzate differenti codici chr$( ). Un esempio può essere:

```
locate 20,12: print chr$(240)
```

Vedrete una freccia al centro dello schermo. Notate che:

20 è la coordinata orizzontale (tra 1 e 40)
12 è la coordinata verticale (tra 1 e 25)
240 è il codice del carattere (tra 32 e 255)

Per riempire un'intera linea con il carattere di codice 250, è sufficiente questo programma:

```
5 cls
10 for x=1 to 39
20 locate x,20
40 print chr$ (250)
50 next
60 goto 5
```

Premete [ESC] due volte per fermare il programma.
Se volete ottenere un semplice effetto di movimento cambiate la riga 40 in:

```
40 print " "; chr$ (250)
```

Al run l'omino verrà mosso lungo tutta la ventesima linea dello schermo. Per dare un po' più di vita al movimento aggiungete:

```
30 call &bd19
```

L'effetto di movimento può essere ulteriormente migliorato aggiungendo cicli di ritardo e sfruttando meglio i caratteri a disposizione. Aggiungete le righe seguenti:

```
60 for n=1 to 300: next n
65 for x=39 to 1 step -1
70 locate x,20
75 call &bd19
80 print chr$ (251); " "
85 next x
90 for n=1 to 300: next n
95 goto 10
run
```

Provate anche l'interessante programmino che segue, senza tener conto del fatto che alcuni comandi saranno spiegati più avanti. Limitatevi a battere:

```
new
10 MODE 1
20 LOCATE 21,14:PRINT CHR$(244)
30 TAG
40 FOR x=0 TO 624 STEP 2
```

```
50 MOVER -16,0
60 IF x<308 OR x>340 THEN y=196:GOTO 90
70 IF x<324 THEN y=x-104:GOTO 85
80 y=536-x
85 SOUND 1,0,20,7
90 MOVE ox,oy:PRINT" ";:ox=x:oy=y
100 MOVE x,y
110 IF (x MOD 4)=0 THEN PRINT CHR$(250); ELSE PRINT CHR$(251);
120 FOR n=1 TO 4:CALL &BD19:NEXT n
130 NEXT x
140 TAGOFF
150 GOTO 20
run
```

PLOT

Il comando PLOT serve per accendere un pixel sullo schermo tramite l'indicazione delle coordinate x,y dello stesso. Ricordiamo che un pixel è il più piccolo elemento singolarmente indirizzabile sullo schermo e che il cursore grafico non è visibile ed è diverso dal cursore normale.

Indipendentemente dal Modo selezionato le coordinate grafiche dello schermo vanno da 0 a 639 sulle x e da 0 a 399 sulle y. L'origine del sistema di riferimento si trova nell'angolo in basso a sinistra dello schermo che ha coordinate 0,0.

Per rendervi conto di ciò resettate il computer con [CTRL] [SHIFT] [ESC] e battete:

```
plot 320,200
```

ottenendo l'accensione del pixel al centro dello schermo.

Cambiate pure modo:

```
mode 0
plot 320,200
```

e vedrete che il pixel acceso è ancora al centro dello schermo, con l'unica variante della dimensione che è lievemente maggiore. Anche con:

```
mode 2
plot 320,200
```

avrete un pixel molto piccolo, sempre al centro dello schermo.

Esercitatevi accendendo punti nei vari Modi e quando avete finito pulite lo schermo e tornate al Modo 1 con:

```
mode 1
```

DRAW

Il comando DRAW traccia un segmento a partire dalla posizione corrente del cursore grafico. Il programma che segue vi permette di tracciare un rettangolo: si parte con il posizionare il cursore grafico tramite il comando PLOT e si prosegue tracciando un primo segmento dalla posizione corrente all'angolo in alto a sinistra, indi di qui all'angolo in alto a destra e così via:

```
5 cls
10 plot 10,10
20 draw 10,390
30 draw 630,390
40 draw 630,10
50 draw 10,10
60 goto 10
run
```

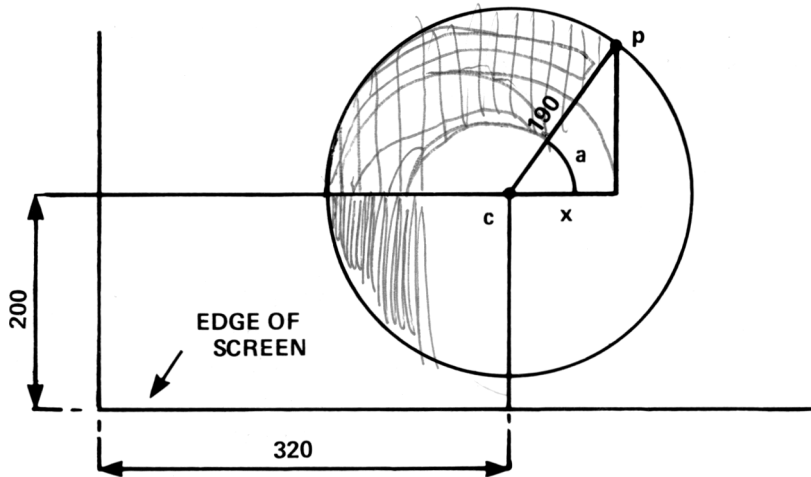
Per uscire premete [ESC] due volte.

Per disegnare un secondo rettangolo dentro il primo aggiungete le righe:

```
60 plot 20,20
70 draw 20,380
80 draw 620,380
90 draw 620,20
100 draw 20,20
200 goto 10
run
```

CIRCONFERENZE

Si possono tracciare circonferenze sia usando il comando PLOT che il comando DRAW. Con il primo metodo definire le coordinate x,y di tutti i punti del cerchio. Riferitevi alla figura sottostante per capire come sia possibile dare le coordinate x,y del punto P che descrive la circonferenza:

x = 190*cos(a)   y = 190*sin(a)

Siccome abbiamo già detto che l'origine degli assi è posta nel punto in basso a sinistra dello schermo, se vogliamo posizionare il centro del cerchio in coincidenza del centro dello schermo dobbiamo dare come coordinate 320 per le x, 200 per le y. Il programma relativo al tracciamento della circonferenza sarà quindi del tipo:

```
new
5  cls
7  deg
10 for a=1 to 360
20 plot 320,200
30 plot 320+190*cos(a),200+190*sin(a)
40 next
run
```

Il raggio della circonferenza può essere ridotto modificando il valore 190 (190 si riferisce al numero di pixel). Cancellando la riga 7 il disegno della circonferenza avverrà in modo differente, a causa della misurazione delle ampiezze angolari in radianti.

Per ottenere il riempimento del cerchio con raggi affiancati l'uno all'altro, sostituite alla riga 30 il comando PLOT con il comando DRAW lasciando il resto inalterato e provate con e senza la riga 7.

Notate che nella riga 40 abbiamo usato semplicemente NEXT invece di NEXT a. Questo è ammesso, in quanto il computer automaticamente associa il NEXT al FOR corrispondente. Naturalmente in programmi che contengono molti cicli FOR/NEXT l'identificazione esplicita della variabile di controllo è utile per una migliore comprensione del listato.

NEW

Abbiamo già avuto modo di usare il comando NEW per ottenere che il computer cancelli completamente la memoria, con effetto simile a quello ottenuto tramite la sequenza [CTRL] [SHIFT] [ESC]. La differenza consiste nel fatto che il NEW non pulisce lo schermo, per cui può essere utile quando volete conservare lo schermo per vostra comodità.

ORIGIN

Nel programma relativo al tracciamento della circonferenza abbiamo utilizzato il comando PLOT per posizionare il centro del cerchio, riferendo le coordinate x, y a questa posizione. Invece di seguire questa procedura, si può usare il comando ORIGIN, con il quale si ridefinisce l'origine del sistema di riferimento. In questo modo le coordinate dei punti della circonferenza sono all'atto pratico riferite al punto 0,0 ridefinito:

```
new
5  cls
10 deg
20 origin 320,200
30 for a=1 to 360
40 plot 190*cos(a),190*sin(a)
50 next
run
```

Anche in questo caso provate a togliere la riga 10 e a sostituire la PLOT con la DRAW.

Il programma che segue disegna sullo schermo quattro circonferenze concentriche:

```
new
5 cls
10 for a=1 to 360
15 deg
20 origin 196,282
30 plot 50*cos(a), 50*sin(a)
40 origin 442,282
50 plot 50*cos(a), 50*sin(a)
60 origin 196,116
70 plot 50*cos(a), 50*sin(a)
80 origin 442,116
```

```
90 plot 50*cos(a), 50*sin(a)
100 next
run
```

GOSUB RETURN

Quando un programma contiene delle righe che devono essere eseguite più volte, è consigliabile raggrupparle in una subroutine, che può essere richiamata dal comando GOSUB seguito dal numero di riga. La fine di ogni subroutine è identificata dall'istruzione RETURN, che rimanda il programma alla istruzione seguente la GOSUB.

Ad esempio, nel programma precedente l'istruzione:

```
PLOT 50*cos(a),50*sin(a)
```

risulta ripetuta quattro volte, mentre, utilizzando una subroutine, avremmo:

```
 10 CLS
 20 DEG
 30 FOR a=1 TO 360
 40 ORIGIN 196,282
 50 GOSUB 140
 60 ORIGIN 442,282
 70 GOSUB 140
 80 ORIGIN 196,116
 90 GOSUB 140
100 ORIGIN 442,116
110 GOSUB 140
120 NEXT
130 END
140 PLOT 50*COS(a),50*SIN(a)
150 RETURN
run
```

Notate l'uso della END alla riga 110, senza la quale il programma passerebbe ad interpretare la riga 120, che invece è richiesta solo dietro richiamo della GOSUB.

A conclusione di questa Sezione vi proponiamo un programma che utilizza la maggior parte dei comandi fin qui esaminati:

```
new
10 MODE 0:BORDER 6:PAPER 0:INK 0,0
20 GOSUB 160:FOR x=1 TO 19:LOCATE x,3
30 PEN 15:PRINT" ";CHR$(238)
```

```
 40 FOR t=1 TO 50:NEXT t:SOUND 2,(x+100)
 50 NEXT x:GOSUB 160:FOR b=3 TO 22
 60 LOCATE 20,b:PEN 7:PRINT CHR$(252)
 70 CLS:GOSUB 160:NEXT b
 80 SOUND 2,0,100,15,0,0,1
 90 GOSUB 160:BORDER 16,24:LOCATE 20,25
100 PEN 14:PRINT CHR$(253);
110 FOR t=1 TO 1000:NEXT t
120 BORDER 6:GOSUB 160:FOR f=3 TO 24
130 LOCATE 10,(25-f):PEN 2
140 PRINT CHR$(144):CLS:GOSUB 160
150 SOUND 7,(100-f),5:NEXT f:GOTO 10
160 LOCATE 10,25:PEN 12
170 PRINT CHR$(239):RETURN
run
```

SUONO

Gli effetti sonori sono generati da un altoparlante interno al computer: quando utilizzate il modulatore/alimentatore MP1 e il vostro TV color, abbiate cura di mettere al minimo il controllo del volume sul televisore.

Il livello del suono può essere regolato agendo sulla rotella marcata VOLUME sulla destra del computer. È anche possibile inviare il suono sulla presa ausiliaria di ingresso di un impianto stereo tramite la presa marcata I/O, che si trova a sinistra sul retro del computer. Questo consente di udire il suono generato dal CPC 464 in stereofonia, su altoparlante o in cuffia.

Il comando SOUND richiede sette parametri, di cui solo i primi due sono obbligatori:

SOUND stato canale, periodo di tono, durata, volume, inviluppo di volume, inviluppo di tono, periodo di rumore.

Negli esempi che seguono utilizzeremo 1 per indicare lo stato canale, in modo da inviare il suono sul canale A.

Periodo di tono

Dall'Appendice VII ricavate come il DO Centrale abbia un periodo di tono pari a 478. Battete:

```
new
10 sound 1,478
run
```

e sentirete un DO Centrale con durata di 1/5 secondi.

Durata

Quando non è altrimenti specificata, la durata di ogni suono è pari a 1/5 secondi. L'unità di misura è 1/50 secondi, per cui occorrerà usare 100 per definire una durata di 1 secondo, 200 per significare 2 secondi e così via. Battendo:

```
10 sound 1,478,200
run
```

sentirete il DO Centrale per 2 secondi.

Volume

Con questo parametro si specifica il volume di partenza di una nota, tramite un numero compreso tra 0 e 7. Se si definisce un inviluppo di volume, il campo di variazione si estende tra 0 e 15. Il valore di default è 4. Battete:

```
10 sound 1,478,200,3
run
```

Fate attenzione al volume di questo suono e provate a definire un livello di volume più alto:

```
10 sound 1,478,200,7
run
```

A voi il confronto.

Inviluppo di volume

L'inviluppo di volume si definisce tramite il comando ENV, che richiede normalmente 4 parametri. Gli ultimi tre parametri possono riferirsi ad ognuna delle cinque sezioni di inviluppo ammesse. Negli esempi useremo una sola sezione, mentre per i particolari si rimanda al Capitolo 6:

ENV numero inviluppo, numero segmenti, ampiezza segmento, tempo di pausa

Numero inviluppo

Si tratta di un numero che identifica un particolare inviluppo. Deve essere compreso tra 0 e 15.

Numero segmenti

Ogni sezione può essere composta da un numero di segmenti compreso tra 0 e 127. Questo parametro è strettamente collegato al tempo di pausa.

Ampiezza segmento

L'ampiezza di ogni segmento è costante nell'ambito della stessa sezione e viene definita da un numero compreso fra −128 e +127. Le variazioni di ampiezza tra l'ultimo segmento di una sezione e il primo della successiva non possono superare le 15 unità.

Tempo di pausa

Questo parametro specifica l'intervallo tra i segmenti in centesimi di secondo con un numero compreso tra 0 e 255. Fate la prova seguente con l'inviluppo di volume:

```
5 env 1,10,1,100
10 sound 1,284,1000,1,1
run
```

La riga 10 definisce un suono con un periodo di tono pari a 284, durata di 10 secondi e volume di partenza pari a 1. L'inviluppo di volume 1, descritto alla riga 5, consiste di 10 segmenti, con incremento di volume fra i segmenti pari a 1 e durata di ogni segmento pari a 1 secondo (100×0.01 secondi). Per avere un'idea immediata di come varia il suono al variare dell'inviluppo, modificate la riga 5 come mostrato di seguito e date il run ogni volta:

```
5 env 1,100,1,10
5 env 1,100,2,10
5 env 1,100,4,10
5 env 1,50,20,20
5 env 1,50,2,20
5 env 1,50,15,30
```

Per finire provate con:

```
5 env 1,50,2,10
```

e notate come il livello rimanga costante a partire da circa la metà della durata del suono e fino alla fine. Questo perchè il numero di segmenti è di 50 e la durata di ogni segmento è di 0.1 secondi. Pertanto, il tempo durante il quale l'ampiezza varia è di soli 5 secondi, ma la durata del suono era stata posta a 10 secondi nel comando SOUND.

Continuate ad esercitarvi da soli per comprendere a fondo il meccanismo.

Inviluppo di tono

Il comando che definisce l'inviluppo di tono è ENT, che richiede normalmente 4 parametri. Vale quanto già detto per l'inviluppo di volume. Per i dettagli riferitevi al Capitolo 6:

ent numero inviluppo, numero segmenti, periodo di tono del segmento, tempo di pausa.

Numero inviluppo

Identifica univocamente un determinato inviluppo, in modo da poter essere richiamato nel comando sound tramite un numero compreso tra 1 e 15.

Numero segmenti

Strettamente connesso al tempo di pausa, è un numero compreso tra 0 e 239. Per esempio, potete definire 10 segmenti di 1 secondo ciascuno.

Periodo di tono del segmento

Il periodo di tono per ogni segmento può variare tra −128 e +127. Segmenti negativi hanno come effetto di aumentare la frequenza della nota, rendendo quest'ultima più alta. Il minimo periodo possibile è 0 e questo deve essere tenuto presente nel calcolo dell'inviluppo di tono. Tutti i dettagli si trovano nell'Appendice VII.

Tempo di pausa

Questo parametro specifica l'intervallo di tempo tra due segmenti in centesimi di secondo con un numero compreso tra 0 e 255. Il tempo di pausa più lungo è quindi pari a 2.55 secondi. Provate con il programma seguente:

```
5  ent 1,100,2,2
10 sound 1,284,200,15,0,1
run
```

La riga 10 definisce un suono con periodo di tono pari a 284, durata di 2 secondi e volume iniziale di 15 (massimo). Non c'è inviluppo di volume (come indica lo 0), mentre l'inviluppo di tono è il numero 1, che viene descritto alla riga 5 come composto da 100 segmenti con incremento del periodo di tono (riduzione della frequenza) pari a 2 ogni 0.02 secondi.

Per ascoltare gli effetti derivanti da variazioni dell'inviluppo di tono, modificate la riga 5 e date il run ogni volta:

```
5 ent 1,100,-2,2
5 ent 1,10,4,20
5 ent 1,10,-4,20
```

Ancora un programmino dimostrativo:

```
5 ent 1,2,17,70
10 sound 1,142,140,15,0,1
15 goto 5
run
```

Per uscire ricordate di premere [ESC] due volte.

A questo punto potete combinare gli inviluppi di tono e volume e il comando SOUND per ottenere svariati effetti sonori:

```
new
5 env 1,100,1,3
10 ent 1,100,5,3
20 sound 1,284,300,1,1,1
run
```

Modificate dapprima la riga 10 battendo:

```
10 ent 1,100,–2,3
```

Indi provate con:

```
5 env 1,100,2,2
10 ent 1,100,–2,2
20 sound 1,284,200,1,1,1
run
```

Continuate ad esercitarvi da soli.

Rumore

È possibile aggiungere del rumore alla fine del comando SOUND con un numero compreso tra 1 e 31. Provate con:

```
5 env 1,100,3,1
20 sound 1,200,100,1,1,1,5
run
```

Cercate ora di ottenere effetti sonori di ogni genere modificando l'inviluppo di volume e il comando SOUND. Provate con e senza rumore.

CAPITOLO 1

PRIMO APPROCCIO

Chi, avendo saltato le istruzioni rivolte ai principianti, nonchè i dettagli sulle connessioni, sull'accensione e sulla familiarizzazione con la tastiera non trovasse la terminologia usata sufficientemente chiara farà bene a rileggere la parte inerente i fondamenti.

Argomenti trattati in questo Capitolo:

— Convenzioni terminologiche usate in questa guida.
— Accensione.
— Tastiera: proprietà.

Indipendentemente dal grado di familiarità che possedete con i computer e la loro programmazione, fate soprattutto attenzione a seguire correttamente le istruzioni di messa a punto. Se avete appena aperto la scatola e non vedete l'ora di cominciare, allora questo Capitolo vi dirà tutto ciò di cui avete bisogno per soddisfare la vostra curiosità iniziale ed entrare così nel vivo delle applicazioni BASIC. Questa sezione intende rivolgersi a quegli utilizzatori che abbiano della familiarità con i computer. È meglio che i principianti partano dal Capitolo introduttivo.

Importante — È d'obbligo che leggiate quanto segue

TERMINOLOGIA

Allo scopo di chiarire i riferimenti ai tasti della tastiera, presenti sia nel testo che nei listati, si sono usate le seguenti convenzioni in tutto il manuale:

[ENTER]: i tasti cui non corrisponde lo stesso carattere/i sullo schermo sono rappresentati in questa forma, contornati da parentesi quadre.

QWERTYUIOP: i tasti che hanno un carattere corrispondente stampato sullo schermo sono rappresentati in questa forma, senza parentesi quadre.

10 FOR N = 1 to 1000: il testo, sia che appaia sullo schermo sia che debba essere battuto sulla tastiera, è presentato in questa forma. Si noti la differenza tra la cifra 0 e la lettera maiuscola O.

Si assume che terminiate ogni programma o linea di programma premendo il tasto [ENTER], perciò questo comando non sarà ripetuto nei listati presentati nel prosieguo di questa guida.

Inoltre si assume che battiate il comando 'run' dopo l'inserimento di ciascun programma. Il BASIC converte tutti i comandi inseriti da minuscolo in MAIUSCOLO quando il programma viene LISTato. Gli esempi che vengono qui presentati sono scritti in MAIUSCOLO in quanto corrispondenti al modo in cui appare il programma quando viene LISTato. Se inserite i programmi usando le minuscole sarete in grado di ritrovare eventuali errori nelle parole chiave con maggiore facilità, poichè gli errori nella battitura delle parole chiave saranno LISTati in minuscolo.

1.1 — Aprite la scatola!

1.1.1 — il monitor a colori

IMPORTANTE

Riferitevi alle istruzioni di messa a punto esposte in modo dettagliato all'inizio di questo manuale d'uso, dove sono descritte le connessioni principali tra le parti del vostro equipaggiamento.

Con il computer correttamente connesso come mostrato nella Figura 1 accendete il monitor e quindi il computer usando l'interruttore situato sul fianco destro della console. Dopo circa 30 secondi di autotest, vedrete comparire la scritta:

```
Amstrad 64K Microcomputer  (v1)

©1984 Amstrad Consumer Electronics plc
          and Locomotive Software Ltd.

BASIC 1.0

Ready
█
```

Questo messaggio è conosciuto con nomi diversi quali: "Reset", "Early Morning", "Wake up", e sta ad indicare che il CPC 464 ha completato il 'reset' (ripristino) allo stato iniziale, condizione che si verifica all'accensione o dopo un reset completo da tastiera, ottenuto premendo simultaneamente la sequenza [CTRL][SHIFT] e [ESC]ape. Provate prima di procedere oltre.

Regolate il controllo di LUMINOSITÀ sul lato destro del monitor. Il colore è preselezionato in fabbrica, pertanto se volete cambiare i colori dello schermo (lettere giallo chiaro su sfondo blu), dovete ricorrere a opportune linee di programma che potete trovare saltando al Capitolo sulla grafica ed al Capitolo sulle parole chiave del BASIC (Cap. 8).

Se proprio siete impazienti, vi presentiamo un breve programma che vi permetterà di selezionare una delle combinazioni testo/sfondo nel modo a 80 colonne. Battete:

```
10 MODE 2
20 INK 1,0
30 INK 0,13
40 BORDER 13
```

Potete anche, in alternativa, inserire le righe di programma di cui sopra nel modo diretto, cioè senza numeri di riga.

Se questo programma non vi dice nulla o se non riuscite a farlo funzionare, sarà meglio che torniate ai Fondamenti posti all'inizio di questa guida.

Troverete che il modo a 80 colonne (MODE 2) è la modalità di testo più utile nello sviluppo dei programmi.

Potreste inoltre voler salvare questo programma (dopo aver letto tutto il Capitolo 2, s'intende !) all'inizio di una cassetta vergine, al fine di non doverlo riscrivere ogni volta che ne avete bisogno.

ATTENZIONE ! l'eccessiva luminosità del monitor potrebbe essere causa di affaticamento visivo, allo stesso modo in cui lo potrebbe essere una regolazione diversa da quella richiesta dalle condizioni dell'ambiente di lavoro. Nel momento in cui percepite un inizio di affaticamento visivo, spegnete tutto e dedicatevi ad un'altra attività, ma per prevenirlo innanzitutto rispettate le seguenti regole:

1 Lavorate sempre in condizioni di luce sufficienti alla lettura dei caratteri di questo manuale. La lettura di questo manuale con il solo ausilio della luminosità dello schermo è assolutamente non consigliata!

2 Usate il minimo di luminosità richiesto per una visione chiara dei caratteri sullo schermo; in modo tale, cioè, da percepire chiaramente ciò che state battendo.

3 Sedetevi il più lontano possibile dallo schermo, ovviamente senza esagerare.

Una lampada da tavolo posizionata a fianco del monitor vi aiuterà a ridurre l'affaticamento visivo, purchè il suo orientamento non provochi riflessi sullo schermo.

1.1.2 — Il monitor a fosfori verdi

Il monitor GT64 dispone di tre controlli posizionati sul frontale sotto lo schermo. Essi servono alla regolazione della LUMINOSITÀ (BRIGTHTNESS), del contrasto (CONTRAST) (che corrisponde alla scarto di luminosità esistente tra le parti chiare e le parti scure rappresentate sullo schermo) e del SINCRONISMO VERTICALE (Vertical HOLD), la cui esatta regolazione evita fastidiose instabilità dello schermo sulla verticale.

La regolazione del SINCRONISMO VERTICALE non richiederà frequenti aggiustamenti. Spesso dopo una corretta regolazione iniziale questo controllo potrà essere tranquillamente dimenticato. La LUMINOSITÀ ed il CONTRASTO dovranno essere regolati di volta in volta sulla base delle condizioni di luminosità esistenti nell'ambiente in cui il CPC 464 sarà usato.

Utilizzando un monitor monocromatico (che è essenzialmente un monitor che dispone di un solo colore per la rappresentazione dei messaggi) si avranno a disposizione gli stessi comandi per la gestione del colore carattere/sfondo e bordo disponibili con la versione a colori; ovviamente anzichè ottenere colori diversi si otterranno diverse gradazioni di verde.

Benchè un uso prolungato del vostro computer con il monitor GT64 possa comportare fenomeni di affaticamento visivo, le vostre sedute di lavoro saranno meno faticose. In particolar modo lo schermo monocromatico vi permetterà un uso pieno del modo di testo ad 80 colonne (80 colonne equivalgono ad una riga contenente 80 simboli alfanumerici) grazie alla risoluzione (che corrisponde all'abilità di rappresentare un numero più alto di elementi primitivi 'pixel' l'uno accanto all'altro senza sovrapposizioni o sfocature). Infatti un monitor monocromatico dispone, generalmente, di una risoluzione superiore a qualsiasi monitor a colori, benchè questi ultimi siano assai più costosi.

La regolazione della luminosità deve essere effettuata in modo tale da fornire un'adeguata visibilità dello schermo, senza che i punti (o pixel) divengano eccessivamente sfocati.

Per selezionare il modo ad 80 colonne (MODE 2), inserite il seguente breve programma nel CPC 464. Esso vi fornirà una selezione tra diversi 'schemi'.

```
10 REM selezione di formati video
20 FOR n=0 TO 26
30 MODE 2
40 INK 1,n
50 INK 0,(26-n)
```

```
55 BORDER n
60 LOCATE 15,12: PRINT "premete qualsiasi tasto alfanumerico per cambiare il formato dello schermo"
70 a$=INKEY$
80 IF a$=" " GOTO 70
90 NEXT
100 GOTO 20
```

Quanto sopra illustra un ulteriore punto concernente lo stile di presentazione di questo manuale. Alcuni listati di programmi andranno a capo (cioè in fase di inserimento oltrepasseranno il fine riga) ed è importante notare che quando si stampano i listati, gli spazi addizionali non sono di per sè richiesti: essi sono aggiunti per ragioni di ordine nei listati stessi.

Il programma non mostra tutte le possibili gradazioni di intensità di verde ottenibili, ma permette comunque una approssimazione sufficiente a valutare le disponibilità di combinazioni di colore. Quando incontrate una combinazione di colore soddisfacente premete il tasto [ESC] due volte per bloccare il programma (apparirà quindi un messaggio di interruzione di programma *break*).

Da qui in avanti il manuale si riferirà quasi sempre alla versione a colori del monitor. Programmi che utilizzano interessanti effetti sul monitor a colori possono essere quasi totalmente mortificati se usati su un monitor monocromatico. Si è comunque prestata una considerevole attenzione a produrre una scala di colori per quanto possibile assai vicina alla scala dei grigi, cui corrispondono le diverse intensità di verde (si veda, per una descrizione più completa, il Capitolo 5).

Il vantaggio del monitor monocromatico risiede soprattutto nelle sue qualità di non affaticamento visivo in fase di sviluppo di programmi, ma se la programmazione non costituisce il vostro principale interesse, il monitor a colori sarà sicuramente un'alternativa irrinunciabile.

1.1.3 — L'alimentatore/modulatore TV esterno MP1

Lo MP1 è un'unità addizionale che potete acquistare nel caso possediate il CPC 464 dotato di monitor a fosfori verdi. Lo MP1 vi permette di utilizzare il vostro computer con TV color domestico consentendovi così di sfruttare a fondo anche la gestione completa del colore del CPC 464.

IMPORTANTE

Riferitevi alle istruzioni di messa a punto esposte in modo dettagliato all'inizio di questo manuale d'uso dove sono descritte le connessioni principali tra le parti del vostro computer.

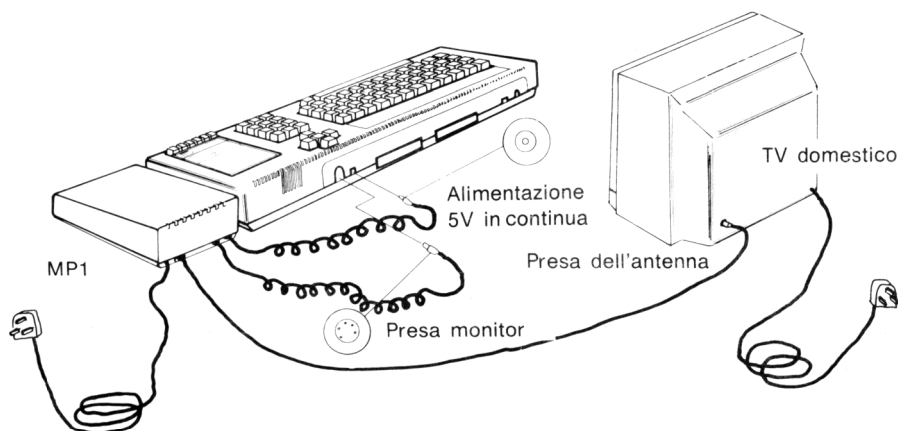



Figura 2 — Connessioni tra lo MP1, il computer e la presa d'antenna del TV color.

L'ALIMENTATORE/MODULATORE TV (MP1) deve essere posizionato alla destra del computer su un supporto adatto vicino al televisore ed alle prese di corrente, come mostrato sopra.

Abbassate al minimo il volume del televisore in quanto il suo fruscio potrebbe disturbarvi nella ricezione del suono proveniente dall'altoparlante proprio del CPC 464. Accendete il TV ed il computer mediante il tasto marcato con la scritta POWER, posizionato sul lato destro.

Il led rosso "ON", situato al centro in alto sul computer, deve essere acceso. Ora dovete sintonizzare il vostro televisore in modo da ricevere i segnali dal computer.

Se avete un televisore con selezione dei canali a pulsante, cercate un canale libero o inutilizzato. Regolate il controllo di sintonia in base alle istruzioni del vostro TV (il segnale si troverà in un intorno del canale 36) finché non ricevete un'immagine simile a quella mostrata di seguito:

```
Amstrad 64K Microcomputer  (v1)

©1984 Amstrad Consumer Electronics plc
          and Locomotive Software Ltd.

BASIC 1.0

Ready
█
```

Regolate il TV finché non percepirete l'immagine nel modo più nitido possibile. La scritta sarà nel colore giallo chiaro su uno sfondo blu scuro, che potrebbe anche variare leggermente in base alle condizioni di regolazione del colore del vostro TV.

Se il televisore dispone di una regolazione a manopola, girate il controllo di sintonia finché l'immagine appare perfettamente stabile. (Di nuovo circa sul canale 36).

Poiché il segnale subisce diverse manipolazioni in termini di modulazione e demodulazione, esso arriverà al televisore in condizioni deteriorate. I risultati non saranno altrettanto buoni come quelli ottenuti utilizzando direttamente l'interfaccia con il monitor. Potrete avvertire come i caratteri nella modalità di testo ad 80 colonne (mode 2) non siano perfettamente nitidi, nel qual caso sarà consigliabile usare il modo a 40 colonne.

1.2 — Primi passi

Ora siete a posto: l'alimentazione risulta connessa, il video è acceso, sia che stiate usando il monitor o il modulatore MP-1, e il CPC 464 attende un input.

Sullo schermo compare il messaggio di "acceso", cioé il messaggio di "wake up", che è il solo testo stabilmente inserito nella memoria ROM del computer visualizzabile senza alcuna istruzione addizionale da tastiera.

Se avete familiarità con il linguaggio di programmazione BASIC ci sono delle buone probabilità che abbiate già inserito un breve programmino "così tanto per familiarizzare". Le parole chiave del BASIC AMSTRAD vi saranno, sotto molti aspetti, note. Comunque, per darvi una spinta, vi proponiamo un breve programma che vi mostrerà il set standard dei caratteri disponibili nella ROM del computer. Questo è il "character set" (set dei caratteri), termine che denota il set complessivo dei caratteri alfanumerici e grafici stampabili sul video a seguito della pressione dei tasti.

Alcuni dei caratteri che vedrete non sono direttamente accessibili da tastiera, ma sono visualizzabili mediante l'istruzione BASIC

```
PRINT CHR$ (<codice ASCII>)
```

descritta più avanti su questo manuale. Questo perché l'unità con cui i dati vengono rappresentati internamente al computer è il "BYTE" (l'elemento atomico è il "BIT"; ogni byte è composto da 8 bit nel caso del CPC 464) e, come potrete vedere scorrendo l'Appendice II, un byte ha 256 possibili combinazioni di valori (2↑8, 2 elevato a 8 possibili sequenze di 0 ed 1). E poiché il computer deve usare, come minimo, un byte — che è la più piccola quantità significativa per il CPC 464 — per ogni carattere immagazzinato, anche voi avete a disposizione tutte le 256 possibili combinazioni, senza dovervi limitare ai circa 96 caratteri di una macchina da scrivere.

Il set standard dei caratteri è un sottoinsieme di quelli disponibili. Esso è denominato nel mondo dei computer con il termine ASCII, acronimo per:

American
Standard
Code for
Information
Interchange

che prima di tutto è un sistema atto a garantire l'univocità del codice relativo alle informazioni scambiate tra computer. È probabilmente il solo aspetto veramente standard nel mondo dei calcolatori e pertanto è utile che acquisiate dimestichezza con tutti gli aspetti della codifica ASCII. L'Appendice III elenca l'insieme dei simboli con la relativa codifica ASCII e i simboli addizionali disponibili sul CPC 464 con i relativi codici numerici.

Alcuni degli altri caratteri non stampabili possono essere visualizzati utilizzando il tasto CONTROL [CTRL] in combinazione con altri simboli della tastiera, ma non provateci subito, perlomeno finché non avrete capito a fondo le funzioni del tasto [CTRL]; potreste, provando delle combinazioni a caso, ottenere più danni che altro.

1.2.1 Tastiera e caratteri

Per vedere da voi stessi come esattamente appaiono i caratteri sullo schermo, scrivete il seguente programma, così da soddisfare la vostra curiosità e tenere in esercizio il CPC 464. Questo programma vi aiuterà anche a prendere confidenza con la semplicità di programmazione e con il fatto che dopo le operazioni di inizializzazione non dovreste incontrare ulteriori problemi "hardware", in quanto il CPC 464 resterà in attesa di istruzioni software.

Per la correzione degli eventuali errori di battitura del programma si rimanda alla Sezione 1.2.5.

Nella battitura del programma che segue non dovete preoccuparvi di usare lettere MAIUSCOLE o minuscole (il maiuscolo si ottiene premendo il tasto [SHIFT] e contemporaneamente il tasto carattere che si vuole ottenere in maiuscolo): il computer eseguirà il programma senza problemi in tutti e due i modi. Dovete separare i comandi usando o gli spazi o i delimitatori appropriati (virgole, due punti etc.) nella posizione mostrata, poiché il BASIC AMSTRAD permette l'uso di parole riservate (completamente elencate nell'Appendice VIII) come nomi di variabili.

La tastiera è mostrata nella figura 4: tutti i tasti sono ridefinidibili, in modo da accontentare chi vuole a tutti i costi una tastiera "italiana". In più, ai tasti del pad numerico possono essere assegnate stringhe di caratteri con le funzioni più svariate, come descritto nel seguito.

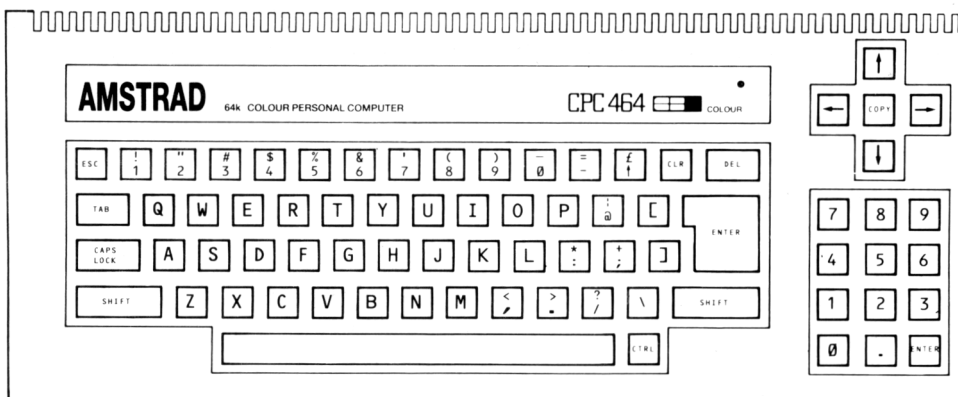



Figura 4 — La tastiera del CPC 464.

La pressione del tasto [ENTER] trasferisce al computer quanto battuto sulla tastiera: può trattarsi di comandi BASIC da eseguire in modo diretto, oppure di righe numerate da memorizzare come parte di un programma. Qualsiasi altra cosa provocherebbe un messaggio di errore.

[ENTER] assume spesso anche la denominazione di "ritorno carrello" o, più semplicemente, "return", con riferimento ai terminali di computer del passato, che si basavano su principi meccanici. Il termine è rimasto, tanto che nel set di caratteri ASCII il codice per [ENTER] viene denotato dalle lettere "CR" (Carriage Return = Ritorno Carrello). Battete:

```
10 FOR N = 32 TO 255
20 PRINT CHR$(N);
30 NEXT N
RUN
```

Osservate cosa succede sullo schermo:

```
Amstrad 64K Microcomputer  (v1)

©1984 Amstrad Consumer Electronics plc
          and Locomotive Software Ltd.

BASIC 1.0

Ready
10 FOR N = 32 to 255
20 PRINT CHR$(N);
30 NEXT N
run
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFG
HIJKLMNOPQRSTUVWXYZ[\]↑_`abcdefghijklmno
pqrstuvwxyz{|}~
Ready
```

Al computer è stato detto di rappresentare sullo schermo l'intero set di caratteri disponibili mediante il breve programma che avete appena scritto. Se non lo avesse fatto andate alla sezione 1.2.7 e vedete come risolvere il problema.

Assumendo che tutto sia andato nel migliore dei modi e che abbiate quindi ottenuto il risultato voluto, esaminiamo che cosa c'è dietro ai caratteri stampati: vi aiuterà a capire come il vostro CPC 464 comunica attraverso il suo set di caratteri.

Per prima cosa occorre notare che non si è data al computer l'istruzione: PRINT "abcdefghilmn....etc."; gli è stato invece chiesto:

```
PRINT CHR$(N)
```

N è un nome, convenientemente corto, assegnato arbitrariamente a una variabile. Per puro caso la lettera N è quella preferita dai matematici, ma qualsiasi altro nome andrebbe bene.

Per variabile intendiamo una informazione che varia a seconda delle istruzioni contenute nel programma. Notate che un numero, come per esempio 5, è fisso (costante) e non è, quindi, una variabile. D'altro canto anche la lettera N è fissa in quanto lettera dell'alfabeto.

Ma allora come mai il computer ha potuto percepire la differenza? Se avessimo voluto dichiarare la lettera N come carattere alfabetico avremmo dovuto racchiuderla tra doppi apici:

```
"N"
```

ed il computer avrebbe risposto con il messaggio Syntax error, poiché non avrebbe identificato come comando la sequenza FOR "N".

Usando semplicemente N come abbiamo fatto, abbiamo appunto comunicato al computer che N identifica una variabile. Per definizione il comando FOR in BASIC richiede di essere seguito da una variabile; così il computer assume che qualsiasi cosa dopo un FOR sia una variabile.

Abbiamo anche comunicato al computer che N = 32 to 255. Così abbiamo delimitato il campo di variabilità tra i due estremi 32 e 255.

Avendo definito la variabile, dobbiamo dire al computer come operare sulla stessa, cosa che facciamo con la riga:

```
20 PRINT CHR$(N);
```

Essa indica al computer di convertire il valore numerico assegnato ad N nel carattere di codice corrispondente: CHR$(N) è la funzione BASIC che effettua questa operazione. Il comando PRINT provoca la stampa su schermo di detto carattere.

Il punto e virgola posto alla fine della riga di istruzioni informa il computer che dopo aver stampato il carattere non deve effettuare nè un CR nè andare a capo. In mancanza del punto e virgola il computer andrebbe automaticamente a capo

e ogni carattere verrebbe stampato su una riga diversa, invece che affiancato al precedente sulla stessa riga.

La linea 30 comunica al computer che dopo aver effettuato l'operazione richiesta con il primo valore della variabile N, in questo caso 32, deve incrementare detto valore di una unità e ripetere l'operazione di stampa sino a quando il valore di N supera 255. Questo processo di iterazione viene definito ciclo e rappresenta uno degli aspetti fondamentali di ogni tecnica di programmazione. Infatti, esso ci risparmia dal dover scrivere lunghe e noiose sequenze di istruzioni strutturalmente identiche, come presto imparerete anche voi fin dai primi programmi.

Dopo che il ciclo ha raggiunto il limite superiore del campo di variabilità, il computer cerca un'istruzione successiva e, non trovando nulla, si ferma e torna al modo diretto stampando Ready per comunicare di essere, appunto, pronto. A questo punto potete introdurre nuovi comandi oppure rilanciare il programma precedente con RUN. Il programma, infatti, è immagazzinato in memoria e vi rimarrà fino a quando non deciderete di cancellarlo oppure di spegnere il computer. In quest'ultimo caso tutti i dati (programma, variabili ecc.) andrebbero persi definitivamente, a meno di non provvedere al loro salvataggio su cassetta.

Questo programma illustra in modo chiaro un punto fondamentale del funzionamento dei computer: qualsiasi operazione eseguita da un computer si riferisce sempre a numeri. Il computer ha rappresentato sullo schermo i caratteri dell'alfabeto ed un certo numero di altri caratteri, utilizzando un numero per identificare il codice del carattere da stampare. Battendo il tasto "A" non chiedete al computer di rappresentare la lettera A sullo schermo, ma di cercare nella propria memoria ROM le informazioni numeriche necessarie a rappresentare la lettera A. L'indirizzo della locazione di memoria dove si trovano tali informazioni è legato al codice numerico generato dalla pressione del tasto.

Ogni carattere dispone di un codice numerico corrispondente: i codici e le corrispondenze sono elencati nell'Appendice III di questo manuale.

Allo stesso modo, la rappresentazione della lettera sullo schermo non ha alcuna relazione con il fatto di "scrivere" un simbolo alfabetico sullo schermo: sempre di numeri si tratta!

1.2.2 Matrice dei caratteri

Non vi spaventate se non capite immediatamente il gergo tecnico usato in questa pagina. È importante rendersi conto di come il computer opera con le vostre istruzioni al fine di giungere ai risultati desiderati, ma è molto probabile che le spiegazioni più tecniche siano ostiche per la maggior parte di voi. Se vi sembra che questo paragrafo sia troppo pesante, passate pure al successivo.

Per cominciare con un esempio, il codice ASCII corrispondente alla lettera A è 97. Il computer, che già non capiva A, non si trova in posizione migliore nei

confronti del codice 97, in quanto anche quest'ultimo deve essere ulteriormente tradotto nell'unico linguaggio che il computer "parla", e cioè un linguaggio binario, per il quale rimandiamo all'Appendice II. In altre parole, la rappresentazione decimale non permette la comunicazione diretta e immediata con il computer, che conosce soltanto sequenze di 0 e 1.

Al primo impatto, le conversioni da notazione decimale a notazione binaria vi appariranno innaturali. Fortunatamente, invece di passare direttamente alla numerazione in base 2, è possibile utilizzare la notazione esadecimale, che ha base 16. Anche se questa conversione non è molto più naturale della prima, non dovrete vedervela con sequenze estremamente lunghe di 0 e 1, nè, dopo un po' di pratica, la cosa vi apparirà troppo complessa.

Dopo aver riconosciuto la pressione del tasto A, il computer esamina la opportuna locazione di memoria e ottiene come risultato un'altra sequenza di numeri che definiscono la matrice del carattere. Questo equivale a dire che ogni carattere rappresentato sullo schermo è costruito su una matrice formata da punti, di cui solo alcuni "visibili". Per ragioni di comodità, nella figura che segue i punti sono sostituiti da quadratini:

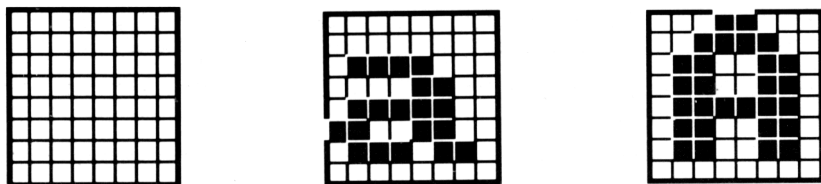



Figura 5 — Le matrici (griglie), rispettivamente da sinistra a destra, di uno spazio, di una lettera 'a' minuscola e di una 'A' maiuscola.

Gli elementi della matrice sono righe e colonne di punti. Il carattere è rappresentato accendendo e spegnendo in opportuna sequenza i punti della matrice. Lo stato di ogni punto (acceso o spento) è memorizzato nella memoria del computer. Le griglie del CPC 464 sono organizzate in una matrice di 8 righe per 8 colonne (ogni riga è definita da un singolo byte; occorrono cioè 8 byte per definire un carattere). Se non trovate tra i 255 caratteri disponibili quello che vi serve, lo potete definire da BASIC mediante le istruzioni SYMBOL e SYMBOL AFTER, il cui uso è descritto nel Capitolo 8.

I caratteri definiti dall'utente possono essere costruiti utilizzando tutte le combinazioni acceso/spento dei 64 punti che compongono la matrice, in modo tale da permettere la creazione di qualsiasi simbolo necessario. Aggiungete a ciò la possibilità di raggruppare diverse matrici per formare blocchi di caratteri e vi accorgerete di come il vostro personale set di caratteri sia limitato solo dalla vostra fantasia e, soprattutto, dal tempo a vostra disposizione.

1.2.3 Ritorniamo al programma

Il risultato del programma che avete appena inserito appare alquanto disordinato: avrebbe senz'altro un aspetto migliore se potessimo pulire lo schermo prima di dare il "RUN". Possiamo, a questo scopo, aggiungere una riga al nostro programma.

Inserite la riga seguente alla posizione attuale dal cursore (il cursore è quel blocco luminoso che appare immediatamente sotto, sulla estrema sinistra, del Ready. In caso di dubbi, sarebbe meglio leggere i FONDAMENTI anzichè questa sezione).

```
5 CLS
RUN
```

Osservate come lo schermo si ripulisca completamente prima di scrivere il set dei caratteri a partire dall'alto a sinistra.

Ciò dimostra anche uno degli aspetti più interessanti del linguaggio di programmazione BASIC: l'ordine mediante il quale inserite le diverse linee di programma non è significativo, poichè l'interprete BASIC le riordina prima di eseguirle in base ad un criterio di numerazione in ordine crescente rispetto al numero di riga. Inoltre, non avete bisogno di vedere l'intero programma rappresentato sullo schermo per inserire nuove righe, in quanto il programma si trova già in memoria.

Le righe di programma vengono automaticamente messe in ordine crescente rispetto al numero prima dell'esecuzione. Provate inserendo l'istruzione LIST.

1.2.4 — LISTing

Potete facilmente controllare qual'è il programma contenuto nella memoria. Inserite:

```
LIST
```

e il risultato sullo schermo è:

```
5 CLS
10 FOR N=32 TO 255
30 NEXT N
Ready
```

Questo programma resterà nella memoria del CPC 464 finchè non farete una delle seguenti operazioni:

- Spegnere il sistema;
- RESET tramite la sequenza [CTRL][SHIFT][ESC];

— LOAD o RUN di un programma da cassetta;
— NEW seguito da [ENTER], che azzera tutte le variabili e pulisce la memoria senza effetti sul Modo di schermo e sui colori.

Ora assegniamo ad uno dei tasti del pad numerico la sequenza: [ENTER]CLS:LIST[ENTER], in modo da sveltire le operazioni di inserimento e sviluppo programmi. Per fare ciò inserite:

```
KEY 138,CHR$(13)+"CLS:LIST"+CHR$(13)
```

Premendo il punto decimale (codice 138) sul tastierino numerico, si otterrà la sequenza desiderata. Si possono ridefinire, in questo modo, fino a 32 tasti selezionabili anche tra quelli della tastiera normale, nel caso voleste usare il pad numerico per l'inserimento di dati numerici. Si veda la descrizione della istruzione KEY nel Capitolo 8.

Se vi trovate ad operare con un lungo programma, definite il tasto come segue:

```
KEY 138,CHR$(13)+"CLS:LIST"
```

e potrete specificare gli estremi del listato, oppure, premendo due volte consecutivamente il tasto ridefinito, ottenere il listing dell'intero programma.

Quando farete degli esperimenti col colore vi potrà capitare di perdervi in una combinazione che non vi permette di leggere le lettere, poichè la penna e lo sfondo hanno lo stesso colore. Per evitare questo inconveniente assegnate, per esempio al piccolo ENTER, la seguente funzione:

```
KEY 139,CHR$(13)+"MODE 2:INK 1,0:INK 0,9"+CHR$(13)
```

A questo punto dovete soltanto premere il più piccolo dei due [ENTER] (quello sul tastierino numerico), per ritornare ad una combinazione di colori che renda visibili le lettere. Il programma in memoria non verrà perso.
Le ridefinizioni dei tasti vengono perse con il reset della macchina. Suggeriamo pertanto di mettere tutto in un programma da salvare su cassetta e da utilizzare quando ritenuto opportuno.

1.2.5 — Primi elementi di Editing

Quando inserirete un programma vi capiterà, inevitabilmente, di commettere errori.
Il CPC 464 è organizzato in modo da rendere la correzione di questi errori la più semplice possibile, evitando allo stesso tempo di sovrapporre nuovi caratteri a quelli che non devono essere corretti.

L'insieme dei tasti che controllano il movimento del cursore vi permette di orientare l'attenzione del computer sulle parti di testo che devono essere modificate.

Quando su una linea che avete inserito viene rilevato un errore, come ad esempio:

10 FOR N=332 TO 255

avete a disposizione diverse opzioni di correzione:

1. Potete battere [ENTER] e riscrivere l'intera linea. La linea contenente l'errore sarà cancellata e sostituita da quella con lo stesso numero.
2. Potete premere il tasto [←] e riportare il cursore sull'errore

Si noti che il carattere sotto il cursore viene rappresentato in negativo. In altre parole, il carattere che normalmente ha lo stesso colore del cursore diventa del colore dello sfondo (o per meglio dire della PAPER), diventando così visibile nonostante che il cursore sia posizionato sopra ad esso.

Ora premete il tasto segnato [CLR] (abbreviazione per CLEAR) ed il carattere sotto il cursore sparirà, mentre la riga verrà compattata in modo da non lasciare vuoti:

10 FOR N=32 TO 255

Premete nuovamente [ENTER] e questa riga, ora corretta, verrà memorizzata come riga di programma. Non è necessario portare il cursore alla fine della riga, in quanto il BASIC accetta la correzione alla pressione dell'[ENTER] indipendentemente dalla posizione del cursore.

3. Potete anche posizionare il cursore sul carattere immediatamante alla destra del carattere che volete cancellare:

10 FOR N = 332 reverse TO 255

Adesso premete il tasto [DEL] (abbreviazione per DELETE=cancella) e il carattere alla sinistra del cursore sarà cancellato, senza influire sul resto della riga.

Alla pressione dell'[ENTER] la riga sarà memorizzata corretta, come già detto.

1.2.6 Considerazioni

I metodi appena esaminati vanno bene se vi accorgete dell'errore prima di aver premuto il tasto [ENTER]. Molti errori, però, non vengono subito notati e ap-

paiono solo durante l'esecuzione del programma. In questo caso il computer risponde con uno dei messaggi di errore elencati nell'Appendice VIII.

Al verificarsi di alcuni errori il computer riscrive automaticamente la riga errata, con il cursore posizionato sulla colonna di estrema sinistra. In questo caso potete usare una delle procedure descritte nel paragrafo precedente per la correzione dell'errore.

Se l'errore non provoca la riscrittura della riga, dovrete listare il programma, cercare l'errore e correggerlo.

1.2.7 Editing con il cursore di "COPY"

Per prima cosa listate il programma mediante il comando LIST. (Continuiamo ad assumere che il programma che stiamo correggendo sia interamente rappresentabile su una sola pagina di schermo).

```
5 CLS
10 FOR N=32 TO 255
20 PRINT CHRS(N);
30 NEXT N
```

L'errore è situato alla riga 20: c'è una S al posto dell'identificatore di stringa $. È possibile riscrivere integralmente la riga 20 o, in alternativa, utilizzare i comandi dell'editor nel modo seguente:

Tenete premuto [SHIFT] (non fa alcuna differenza se è quello situato sulla destra o sulla sinistra) e premete il tasto [↑].

Ad ogni pressione del tasto il cursore si sposterà verso l'alto di una riga. Tenendo il tasto premuto in maniera continua, il comando verrà automaticamente ripetuto fino al rilascio. Se andate oltre la posizione desiderata tornate con il cursore in basso [↓], tenendo sempre premuto il tasto [SHIFT].

Questa azione causa lo sganciamento del cursore di COPY da quello principale (ambedue hanno lo stesso aspetto), e il suo trasferimento sulla riga alla quale siete posizionati per correzioni e modifiche.

Iniziate con il cursore di COPY posizionato sul primo carattere della riga.

```
5 CLS
10 FOR N = 32 TO 255
20 PRINT CHRS (N);
30 NEXT N
Ready
■
```

Se aveste portato il cursore principale sulla riga senza premere lo [SHIFT], il computer non avrebbe accettato la correzione.

Se, dimenticando questa norma, vi doveste ritrovare nella situazione appena descritta, potete rimediare premendo [ESC] e solo DOPO uno dei tasti di [ENTER] o qualsiasi altro tasto di funzione cui sia stata assegnata la stringa CHR$(13). Fate attenzione a non inserire e mandare in esecuzione il comando NEW che causerebbe la perdita dell'intero programma.

Se avete iniziato a scrivere su di una linea e terminate l'inserimento senza premere [ENTER], il computer emetterà un beep per segnalare l'irregolarità dello spostamento. Potete uscire da questa situazione premendo [ENTER], senza che nulla del programma vada perso. Nel caso in cui abbiate inserito un numero di riga valido quale primo carattere, la riga di programma verrà sovrascritta ed occorrerà riscriverla.

Con il cursore di COPY posizionato correttamente cominciate a premere il tasto [COPY] fino a raggiungere il carattere che desiderate modificare. Quando vi sarete fatti un po' di pratica con la velocità con cui il cursore di COPY si sposta potrete sfruttare anche per esso la funzione di autorepeat, disponibile per ogni tasto.

```
5 CLS
10 FOR N = 32 TO 255
20 PRINT CHRS (N);
30 NEXT N
Ready
20 PRINT CHR■
```

Ora rilasciate il tasto [COPY] e battete $, che apparirà sotto il cursore PRINCIPALE, causando lo spostamento dello stesso di un posto verso destra.

```
20 PRINT CHR$ ■
```

Per muovere il cursore di COPY oltre la S da correggere tenete premuto il tasto [SHIFT] e quindi premete il tasto [→] una volta. Il cursore di COPY si fermerà quindi su (. Rilasciate lo [SHIFT] e premete il tasto [COPY] fino a quando non giungete a fine riga. Premete [ENTER] e la linea di programma corretta visualizzata in basso a sinistra sullo schermo rimpiazzerà quella sbagliata mostrata nel listato.

Potete combinare questi diversi modi copiando, tramite [COPY], l'intera riga sulla parte bassa dello schermo per correggerla, prima di battere [ENTER], utilizzando il cursore principale, che sposterete con i tasti abituali. Tutte le funzioni di editing già descritte sono utilizzabili, comprese [CLR] e [DEL] non accompagnate dalla pressione del tasto [SHIFT]. Premendo contemporaneamente i tasti

[CTRL] e [←] o [→], sposterete il cursore, rispettivamente, all'inizio o alla fine della riga che state correggendo.

Con un po' di pratica diverrete presto abili ed eseguirete queste operazioni in pochissimo tempo.

Oppure, infine, potete editare la riga da correggere o modificare battendo:

EDIT 20

Il computer risponde con:

20 PRINT CHRS(N); (cursore sul 2)

A questo punto correggete ciò che dovete spostandovi con il cursore ed utilizzando i tasti [CLR] e [DEL] e quando sarete soddisfatti del risultato premete [ENTER]. Se incappate in qualche guaio premete [ESC] e fate un nuovo listato. La linea da cui siete usciti con [ESC]ape rimarrà immutata; ripetete nuovamente la sequenza di cui sopra finchè la riga non risulterà essere corretta.

Fino ad ora abbiamo sempre lavorato nel Modo di base del CPC 464. Per dare un'occhiata agli altri Modi inserite

MODE 0
RUN

Osservate che lo schermo dopo essersi ripulito mostra 20 larghi caratteri su ogni linea sullo schermo:

Per tornare allo schermo originale scrivete:

MODE 1
RUN

E siamo così ritornati al punto di partenza. Per vedere lo schermo nel Modo a 80 colonne, scrivete:

```
MODE 2
RUN
```

```
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]↑_`abcdefghijklmno
pqrstuvwxyz{|}~
Ready
```

Siamo partiti. Ora dovreste aver soddisfatto almeno alcune delle curiosità iniziali. Gli utenti già esperti avranno ormai cominciato a convertire i loro programmi preferiti in modo tale da poterli eseguire nel dialetto BASIC di cui dispone il CPC 464; gli utilizzatori alle prime armi con il BASIC dovranno passare attraverso la sezione dei concetti di base per avere un'idea delle funzioni specifiche di questa macchina.

CAPITOLO 2
REGISTRATORE DI CASSETTE DATACORDER

Caricamento e operazioni sul registratore di dati incorporato nel CPC 464

Argomenti trattati in questo capitolo:

- Somiglianze e differenze tra cassette audio e cassette dati.
- Caricamento ed esecuzione di programmi su cassetta.
  - la cassetta di presentazione.
- Velocità di registrazione e lettura.
- Salvataggio di programmi su cassetta.
- Errori di lettura.

La memoria RAM del CPC464 è in grado di conservare le informazioni per tutto l'arco di tempo in cui l'alimentazione è connessa e il computer acceso. In altre parole, la memoria RAM si definisce volatile. Se volete salvare il contenuto della memoria per usi futuri dovete provvedere, prima di staccare la spina, a registrare il programma (ed eventualmente anche le variabili) su una cassetta. La cassetta è infatti un supporto di memorizzazione permanente, così come il sistema a dischi collegabile al CPC464.

2.1 — Controlli del registratore

Alla destra della tastiera si trova il registratore di cassette Datacorder, mostrato in figura 2.1. La meccanica del Datacorder è essenzialmente la stessa di un qualsiasi registratore di cassette, eccezion fatta per l'elettronica, ottimizzata per l'uso dello stesso come unità di memorizzazione di massa per computer.

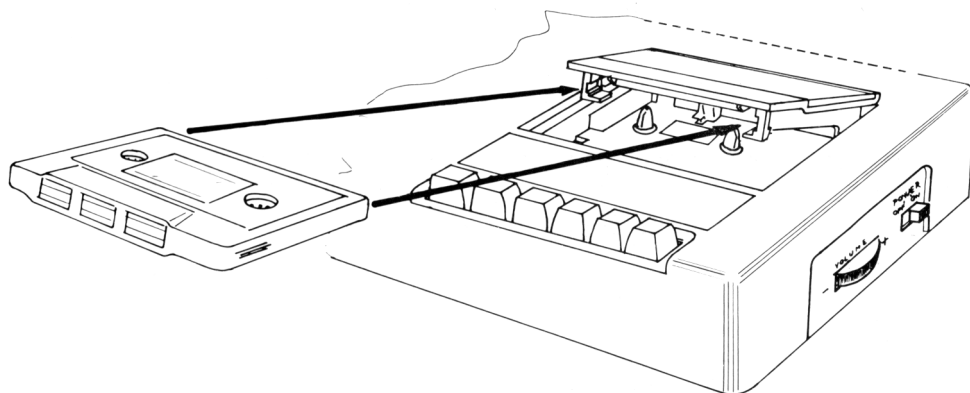



Figura 2.1 — Questo è il giusto modo di inserire la cassetta nel registratore di cui il CPC464 è dotato.

La disposizione dei tasti di controllo del Datacorder è la stessa degli altri registratori.

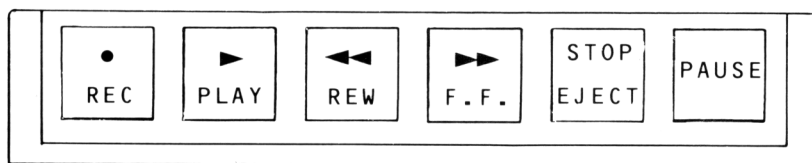



Figura 2.2 — I tasti di controllo del Datacorder

Si noti che questi tasti richiedono una pressione maggiore di quella richiesta dalla tastiera.

[REC] = registrazione (deve essere premuto insieme al tasto [PLAY] per registrare dati quando richiesto da programma). Non è possibile premere il tasto quando la cassetta è priva delle due linguette protettive (vedi figura 2.3), oppure se lo sportello del Datacorder è aperto. Per attivare la funzione dovete premere [REC] fino a quando non rimane in posizione abbassata e quindi il tasto [PLAY]. A questo punto il computer è in grado di iniziare a scrivere sulla cassetta secondo le istruzioni ricevute dal programma o da un comando SAVE introdotto direttamente da tastiera.

[PLAY] = Lettura da cassetta in risposta ad un comando LOAD o RUN". Il computer leggerà i dati da cassetta secondo le istruzioni ricevute dal programma o da un comando diretto. I tasti [REC] e [PLAY] vengono disattivati in modo automatico quando si raggiunge la fine del nastro.

[REW] = Riavvolgimento del nastro dalla bobina di destra a quella di sinistra. Non esiste per questa funzione alcuna forma di disattivazione automatica a fine nastro. Fate quindi attenzione a non dimenticare il nastro in riavvolgimento, in quanto l'eccessiva tensione potrebbe provocare danni.

[F.F.] = Avanzamento veloce dalla bobina di sinistra a quella di destra. Anche [F.F.] non dispone di un sistema di disattivazione automatica e pertanto valgono le stesse avvertenze date sopra per [REW].

[STOP/EJECT] = Blocco delle operazioni sul Datacorder e ripristino delle condizioni originali per tutti i tasti. Premendo e rilasciando il tasto si ottiene semplicemente di fermare il registratore; la successiva pressione provoca l'apertura del coperchio e permette l'accesso alla cassetta. La cassetta non può essere estratta se non in questa condizione.

[PAUSE] = Pausa meccanica ottenibile con il Datacorder nello stato [PLAY] o [PLAY-REC]. L'uso di questo tasto durante operazioni di lettura e/o scrittura darà luogo ad una segnalazione di errore e l'operazione dovrà essere ricominciata da capo. Tutte le operazioni di pausa del Datacorder sono gestite da software rendendo così questo tasto di scarso utilizzo.

2.2 — Protezione delle cassette dalla scrittura accidentale

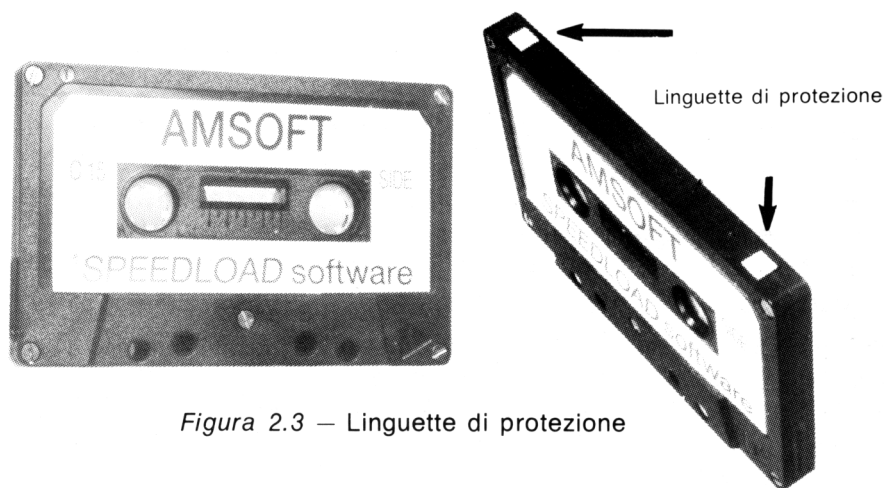



Figura 2.3 — Linguette di protezione

Allo scopo di proteggere i dati immagazzinati sulla cassetta contro cancellazioni casuali dovete asportare le linguette di plastica che si trovano abitualmente sul fondo di tutte le normali compact cassette. Eliminando queste linguette risul-

terà impossibile premere il tasto [REC], proteggendo così la cassetta da altre scritture accidentali che provocherebbero la perdita o il danneggiamento dei dati già immagazzinati.

La protezione dalla scrittura può essere effettuata separatamente per ciascuno dei lati della cassetta; ad ogni lato corrisponde una linguetta. Se in seguito vorrete riabilitare la scrittura su cassetta dovrete coprire le aperture lasciate dalle linguette asportate con del nastro adesivo.

2.3 — Caricamento dati da cassetta

La figura 2.1 illustra il modo corretto di inserire nella macchina il nastro fornito in dotazione con il CPC464.

La cassetta dovrà essere completamente riavvolta, fino al raggiungimento dell'inizio del nastro. Nel caso in cui il nastro dovesse accidentalmente uscire dalle apertura della cassetta, vi consigliamo di riavvolgerlo manualmente prima di inserire la cassetta nel Datacorder. L'uso di una cassetta nelle condizioni appena citate potrebbe provocare la perdita delle informazioni che vi sono immagazzinate.

Vi accorgerete che mentre una cassetta audio può, benchè danneggiata, essere ascoltata su un registratore, una cassetta dati danneggiata non può essere riutilizzata o, quanto meno, non potranno essere recuperati i dati registrati.

Nel caso vi capitasse una fuoriuscita del nastro e la cassetta risultasse ancora funzionante vi consigliamo di resistere alla tentazione di riutilizzare detta cassetta e di provvedere a salvare dati e programmi su una cassetta vergine.

2.4 — Caricamento ed esecuzione della cassetta di presentazione

Il nastro fornito con il CPC 464 contiene una serie di dimostrazioni delle capacità grafiche e sonore del CPC 464, del software di base di cui esso è dotato, del sistema operativo e del BASIC AMSTRAD.

Tra le funzioni del Sistema Operativo (MOS Machine Operating System) rientrano anche quelle di gestione della cassetta, incluse quelle di LOAD e RUN che sono sicuramente le più usate.

Il CPC 464 dispone di alcune funzioni che rendono semplice l'inserimento di comandi da tastiera anche ai meno esperti. Se avete appena acceso il computer e state contemplando il messaggio di READY, inserite il nastro come mostrato nella figura 2.1, azzerate il contanastro e premete contemporaneamente i tasti

[CTRL] e il piccolo [ENTER] sul pad numerico. Il computer vi risponderà con il messaggio:

RUN"
Press PLAY then any key:

Per vostra informazione questo è un esempio di ridefinizione dei tasti con assegnazione di una sequenza di comandi ad un unico tasto, in questo caso effettuata automaticamente dal software di base della macchina; ricordiamo che questo argomento è stato brevemente trattato nel Capitolo 1 a proposito del comando KEY. Avreste anche potuto inserire da tastiera il comando RUN" e di seguito premere il tasto [ENTER], ma è più semplice premere due soli tasti anzichè scrivere una riga di programma per ottenere gli stessi risultati.

Nella maggior parte dei casi i due tasti [ENTER] sono equivalenti, il piccolo [ENTER] può però essere ridefinito per eseguire altre istruzioni assegnategli dall'utente. L'argomento delle funzioni assegnabili dall'utente ai tasti verrà trattato più avanti.

Il tasto [PLAY] che occorre premere è quello del Datacorder. Premetelo finchè non resta in posizione abbassata. La parte rimanente del messaggio rappresentato sullo schermo è un modo di dire convenzionale, che potrebbe anche generare confusione negli inesperti. Essenzialmente, serve per proseguire nelle operazioni senza bisogno di ulteriori istruzioni specifiche.

Sarebbe stato più corretto richiedere la pressione di un qualsiasi tasto esclusi [CTRL][CAPS LOCK][SHIFT][ESC] e quelli di controllo del Datacorder, ma è inevitabile che, per ragioni di sinteticità, alcune cose vengano date per scontate assumendo che l'utente sia già stato in qualche modo informato. Ovunque utilizzato in questo manuale, e nel software AMSOFT, vale per tale messaggio quanto appena detto.

La pressione del tasto non produrrà alcun effetto sullo schermo, ma provocherà l'avvio del Datacorder e l'inizio dello scorrimento del nastro. Se il nastro non dovesse partire premete nuovamente un tasto (è buona abitudine usare l'[ENTER] grande) e controllate che non sia premuto il tasto [PAUSE].

Anche premendo uno o più tasti durante le operazioni su nastro non si ottiene alcun effetto, in quanto il controllo è stato trasferito dalla tastiera alle routine di gestione del Datacorder.

Nel caso in cui non venga specificato il nome del programma da caricare, e quindi nessun carattere segua l'istruzione:

RUN"

il computer caricherà il primo programma incontrato sul nastro. Una volta trovatolo, sullo schermo apparirà il messaggio:

Loading WELCOME 1 block 1

Il computer vi avvisa che ha incontrato il primo della serie dei blocchi di dati che compongono il programma denominato "WELCOME 1". Ogni programma è salvato sul nastro a blocchi di dati (lunghi 2Kbyte) che poi verranno letti con lo stesso formato dal computer ; ogni blocco è identificato separatamente sul nastro ed il messaggio sullo schermo vi indica volta per volta il numero progressivo del blocco letto. Dopo la lettura di ogni singolo blocco dati il nastro si ferma momentaneamente per poi ripartire, segnalando il nuovo numero di blocco.

Se in un qualsiasi momento il computer rileva degli errori di lettura, invierà sullo schermo un messaggio di errore (i messaggi di errore sono elencati nell'Appendice VIII) che vi avvisa della natura del problema. Non c'è, a questo punto, altra possibilità se non quella di ricominciare da capo tutta l'operazione finchè il programma non sarà correttamente caricato senza riscontrare errori di lettura.

Assumendo che il vostro programma venga caricato in modo completo, leggete le istruzioni sullo schermo e il vostro nastro "Welcome" farà il resto.

2.5 — Supersafe e Speedload

Il CPC464 offre all'utilizzatore due velocità per le operazioni di lettura/scrittura sul Datacorder: la velocità Supersafe di 1000 baud (1 baud equivale ad 1 bit per secondo) e Speedload di 2000 baud. Quindi la velocità Speedload è doppia di quella Supersafe, anche se occorre dire che la velocità va a scapito della sicurezza, soprattutto quando vengono usati nastri di bassa qualità e quando si presentano problemi dovuti al diverso allineamento delle testine di lettura/scrittura di registratori differenti.

Per programmi registrati e letti sulla stessa macchina lo Speedload è sicuro a patto di usare nastri di buona qualità. Speedload è inoltre capace di leggere direttamente i programmi in commercio su nastro, anche se la AMSOFT consiglia i produttori di registrare i programmi ad entrambe le velocità. Il computer, in lettura, si adatterà automaticamente alla velocità con cui il programma è stato registrato. Siccome la velocità più bassa è selezionata per default, volendo salvare un programma con l'opzione Speedload occorre dichiararlo esplicitamente con il comando:

```
SPEED WRITE 1
```

Per tornare alla velocità di default potete resettare il computer (perdendo tutti i dati), oppure battere:

```
SPEED WRITE 0
```

2.6 — Registrazione di dati e programmi su cassetta

Il BASIC dispone di diversi comandi inerenti il modo in cui le informazioni possono essere registrate o lette sulla cassetta. Le riassumiamo brevemente assieme ad alcuni esempi.

2.6.1 SAVE "<nome file>"

Il modo abituale di registrare i dati su cassetta consiste nell'usare il comando SAVE in modo diretto. Utilizzando il listato del breve programma che rappresenta sullo schermo i caratteri disponibili presentato nel Capitolo 1 possiamo avere, mediante le istruzioni che seguono, una esemplificazione del funzionamento del comando SAVE. Il <nome file> può essere lungo fino a 16 caratteri e includere spazi. Se tentate di scrivere un nome più lungo di 16 caratteri sappiate che dal diciasettesimo in poi esso sarà troncato. Con il programma nella memoria del computer scrivete:

SAVE "CARATTERI"

ed il computer risponderà con il messaggio:

Press REC and PLAY then any key:

Si ricordi l'avvertenza data precedentemente sull'espressione "...any key:". Il nastro partirà ed il computer salverà il programma con il nome 'CARATTERI'.

IMPORTANTE

NB: il computer non può controllare se avete premuto il tasto corretto del Datacorder. Perciò, premendo soltanto [PLAY] il nastro comincerà ad avvolgersi senza registrare e, anche se tutto sembrerà andare bene, il programma non verrà in realtà salvato.

ATTENZIONE!!: se, accidentalmente, premete [PLAY] e [REC] quando invece volete caricare/eseguire un programma da nastro otterrete come risultato la cancellazione delle informazioni contenute nella cassetta fino a quando non interrompete l'operazione premendo [ESC]. Se temete di perdere una parte di dati a causa di un simile errore, abituatevi a togliere le linguette di protezione dalla cassetta.

Oltre a quella appena descritta, esistono altre tre forme per la SAVE che necessitano di spiegazioni più dettagliate.

2.6.2 SAVE "<nome file>",A

La procedura è quella già vista. Con l'aggiunta del suffisso, A si indica al computer di salvare i programmi e/o i dati sotto forma di un file di testo in formato ASCII, piuttosto che nella notazione abbreviata normalmente usata. Questo metodo di registrazione dei dati è applicabile soprattutto a files creati da wordprocessor e da altri programmi applicativi, il suo uso sarà comunque ulteriormente discusso quando se ne presenterà la necessità.

2.6.3 SAVE "<nome file>",P

Il suffisso, P indica al computer di proteggere i dati registrati su cassetta in modo che il programma non possa essere facilmente listato interrompendone l'esecuzione con il tasto [ESC]. I programmi registrati con questa modalità potranno essere richiamati soltanto usando i comandi RUN e CHAIN. Poichè l'opzione, P non è annullabile, raccomandiamo di salvare una copia del programma in forma non protetta in previsione di ulteriori modifiche dello stesso.

2.6.4 SAVE "<nome file>", B, <indirizzo di partenza>, <lunghezza> [,<indirizzo di esecuzione>]

Questa opzione vi permette di salvare una zona della memoria RAM in forma binaria, cioè esattamente come è memorizzata. Occorre però indicare al computer l'indirizzo di inizio del blocco da salvare, la sua lunghezza, nonchè l'indirizzo da cui iniziare l'esecuzione nel caso in cui si tratti di un programma in codice macchina.

Tramite questa opzione è possibile salvare uno screen, o parte di esso, direttamente su cassetta. Uno degli usi più comuni consiste nella creazione di titoli che interrompono la monotonia del processo di caricamento di programmi piuttosto lunghi.

2.6.5 File senza nome e il comando CAT

È ammesso salvare un file senza alcun nome nella forma:

SAVE " "

Sulla casetta è possibile salvare in ordine sequenziale più file dello stesso nome (inclusi quelli senza nome), al contrario di quanto avviene per i dischi dove ogni file deve essere univocamente identificato. Questa prassi è comunque del tutto sconsigliabile, in quanto molto presto vi porterebbe a grosse confusioni tra i vari programmi. Dare un nome ad ogni programma è cosa molto utile, specie se il nome è seguito da un codice numerico ad indicare la data dell'ultimo aggiornamento.

Potete anche fare un elenco (CAT) dei files presenti sulla cassetta digitando il comando CAT seguito da [ENTER] e seguendo poi le istruzioni:

Press PLAY then any key:

Il BASIC visualizzerà il nome di ogni programma incontrato, ciascuno seguito dal numero di blocchi e da un indicativo del tipo di file:

$ se è un programma BASIC standard
% se è un programma BASIC protetto in lettura

* se è un file di testo in codifica ASCII
& se è un file binario

Un OK alla fine della linea vi comunicherà che il file è leggibile e che se aveste richiesto un LOAD di quel programma il computer l'avrebbe caricato senza problemi. L'esecuzione del comando CAT non influenza il programma residente in memoria.

2.7 — Errori di lettura

Gli errori di caricamento vengono segnalati dal messaggio "Read error". Anche dopo il rilevamento di un errore il nastro continuerà a scorrere, i blocchi successivi verranno letti, ma non portati in memoria, a meno che non si tratti del blocco 1 del programma in cui si è verificato l'errore.

Ciò significa che dopo un errore di lettura sarà possibile riavvolgere il nastro e premere [PLAY] senza dover resettare il computer che ritenterà di leggere il programma in cui è stato rilevato l'errore e, con un po' di fortuna, il nuovo tentativo avrà successo.

Gli errori di lettura sono imputabili a molteplici cause, la più comune delle quali consiste nel surriscaldamento o stiramento del nastro. Essi possono anche derivare da un'abitudine all'apparenza innocente quale lo spegnimento o l'accensione del computer quando i tasti [PLAY] o [REC][PLAY] sono in posizione abbassata. In questo caso infatti le testine di lettura/scrittura esercitano una pressione sul nastro e possono essere attraversate da un breve impulso elettrico all'atto della accensione/spegnimento. Anche se il nastro è rimasto fermo, questo effetto può effettivamente cancellare la parte di nastro interessata rendendo così illeggibili i dati che vi sono immagazzinati. Inoltre il nastro, in queste condizioni, rimane fermo stretto tra il capstan e l'alberino di trazione, cosa che, se prolungata, può a sua volta provocare danni.

Altri errori di lettura possono derivare dalla pressione involontaria del tasto [PAUSE] durante le operazioni di lettura/registrazione, oppure dal differente allineamento delle testine di diversi Datacorder.

Questi tipi di errori possono anche avere origini meno identificabili. Le compact cassette non sono state inventate per l'immagazzinamento di dati digitali e sono state adottate sui personal computer soprattutto a causa degli alti costi di sistemi più "professionali". Le limitazioni della cassetta sono soprattutto fisiche ed in particolar modo dovute alla non perfetta magnetizzazione del nastro insieme alla velocità relativamente bassa con cui il nastro scorre sotto le testine. Qualsiasi tentativo di aumentare la velocità oltre certi limiti andrà a scapito dell'affidabilità, specialmente con cassette duplicate su impianti di bassa qualità.

Si noti che le cassette contenenti programmi per altri computer non sono leggibili dal CPC464 anche se l'audio di queste può in qualche modo assomigliare a

quello del CPC464. I programmi di altri computer devono essere adattati e riscritti per funzionare sul vostro computer; è quindi opportuno che prima di tentare manovre che potrebbero anche danneggiare la macchina in vostro possesso ci contattiate.

2.8 — Considerazioni sulle cassette

Sebbene il Datacorder accetti tranquillamente anche cassette da 90 minuti (C 90), è meglio che usiate, soprattutto per ragioni di comodità, cassette C 12 (6 minuti per lato) o, al massimo, C 30. Infatti, i programmi registrati alla fine di cassette lunghe potrebbero essere difficili da rintracciare anche se magari siete disposti a lunghe attese o siete meticolosi utilizzatori del contanastro. Se volete sovrascrivere un programma registrato su una cassetta lunga, è necessario che facciate molta attenzione a riposizionare la testina di lettura/scrittura all'inizio del file che lo contiene per non sovrascrivere altri programmi che verrebbero di conseguenza danneggiati.

Soprattutto cercate di registrare il minor numero possibile di programmi per cassetta; le cassette da 10 minuti sono relativamente poco costose e nel caso di avaria i programmi eventualmente persi sarebbero pochi.

Infine ricordate che il software commerciale è venduto sotto vincoli di copyright piuttosto forti. Non potete duplicare il software fornito se non come da condizioni di vendita di detto materiale. La tentazione di fare copie, anche se solo per amici, è sempre molto forte. Il costo di produzione del software è sempre molto elevato e quindi il prezzo cui viene venduto è compensativo di un'impresa commerciale che si sforza anche di farvi trovare cose sempre nuove e, possibilmente, migliori. Una cassetta copiata difficilmente viene sostituita, anche se il software originale conteneva errori.

CAPITOLO 3
INTRODUZIONE AL BASIC

Una breve introduzione alla programmazione con il BASIC AMSTRAD.
Argomenti trattati in questo capitolo:
- Le regole sintattiche
- Il comando PRINT, i canali e l'organizzazione dello schermo
- Le ZONE di stampa

3.1 — Fondamenti del BASIC

Il legame esistente fra il BASIC del CPC 464 e il funzionamento interno del computer è descritto nell'Appendice II. Se non avete mai programmato un computer vi potrà essere necessario leggere uno dei tanti testi introduttivi disponibili sul mercato, in quanto, nonostante sia nostra intenzione risultare molto semplici e chiari, dovremo, per brevità, dare per acquisite alcune nozioni di base.

Dovreste essere in grado di comprendere tutti gli argomenti trattati in questo capitolo semplicemente seguendo gli esercizi, senza preoccuparvi di entrare nei dettagli di ciò che accade di volta in volta.

Il BASIC è il linguaggio che viene fornito con il CPC 464; essendo contenuto all'interno del computer, è subito disponibile all'accensione. Il BASIC vi avverte della sua presenza con il messaggio:

```
Ready
```

Il BASIC è il linguaggio più semplice da imparare. È composto di parole ben definite, ha una 'grammatica' semplice e opera in modo assolutamente logico.

I comandi del BASIC AMSTRAD sono elencati nel Capitolo 8. Ogni comando è identificato da una o più parole chiave e può richiedere un certo numero di parametri, alcuni dei quali opzionali. In generale ogni parametro può essere visto co-

me una espressione, comprendente costanti, variabili e funzioni. Le combinazioni di numeri e lettere vengono chiamate stringhe. I dati numerici possono essere espressi in forma decimale, esadecimale e binaria.

I file su cassetta vengono gestiti sequenzialmente; non è cioè possibile leggere il decimo file presente sulla cassetta senza far scorrere tutto il nastro dall'inizio.

3.2 — Struttura di un programma BASIC

Un programma BASIC è composto di righe. Ogni riga può contenere più comandi, separati fra loro dai due punti, ma non deve essere lunga più di 255 caratteri. Per 'carattere' si intende un numero, una lettera, un segno speciale o uno spazio. Esistono due modi di inserimento ed esecuzione dei comandi BASIC: il Modo Diretto e il Modo Programma. Nel primo i comandi vengono battuti da tastiera ed eseguiti immediatamente alla pressione del tasto [ENTER]. In Modo Programma i comandi vengono inseriti in righe numerate che poi saranno interpretate ed eseguite in sequenza, secondo quanto indicato dai numeri di riga.

Il BASIC AMSTRAD permette, in Modo Diretto, di aggiungere o togliere righe al programma e di modificare quelle già esistenti. Prima di eseguire i comandi RUN o LIST, il BASIC riorganizza internamente l'ordine delle righe di programma in sequenza ascendente secondo i numeri di riga, senza tenere conto dell'ordine di inserimento.

3.3 — Inserimento delle righe di programma

Il BASIC accetta righe di lunghezza massima pari a 255 caratteri: la fine della riga è indicata dalla pressione del tasto [ENTER]. Una riga di programma può essere modificata anche in fase di inserimento, così come si può utilizzare la funzione COPY per inserire nella riga caratteri provenienti da qualsiasi punto dello schermo (vedere il Capitolo 1 Sezione 1.2.7).

Tutte le istruzioni devono essere delimitate da un separatore, che può essere uno spazio, un operatore matematico (+ — ecc.) o un altro dei caratteri riconosciuti come separatori. Le istruzioni possono essere inserite sia in lettere minuscole che maiuscole.

Il comando PRINT può essere abbreviato con il punto interrogativo [?], senza richiedere, in questa forma, l'uso di separatori. Consigliamo comunque di prendere la buona abitudine di utilizzare sempre lo spazio come separatore, anche quando nella riga sono presenti operatori matematici.

Il singolo apice ' ([SHIFT] 7) è l'abbreviazione del comando REM nelle righe di commento.

Spazi non necessari eventualmente inseriti nel listato del programma non vengono considerati dal BASIC, ma possono essere utilizzati per rendere più leggibile il listato stesso.

3.4 — Terminologia

Nella descrizione delle istruzioni del BASIC ci siamo attenuti ad un formalismo molto preciso, che deve sempre essere tenuto presente. Tutti i parametri sono racchiusi tra i simboli < > e quelli opzionali sono ulteriormente inclusi fra parentesi quadrate [ ]. Per esempio, più volte nella descrizione delle istruzioni si fa riferimento ad un numero, che viene rappresentato come:

<espressione numerica> se obbligatorio

oppure

[<espressione numerica>] se opzionale

Qualora la parte opzionale possa apparire più volte, verrà posto un asterisco dopo la parentesi quadrata di chiusura. Per esempio, una stringa di cifre composta almeno da una cifra, verrebbe descritta come:

<cifra>[<cifra>]*

Una simile espressione potrebbe essere:

34 oppure
344 oppure
345678 ecc.

Ogni cosa non racchiusa tra < > deve essere inserita così come è scritta.
Una lista di elementi separati da virgole viene indicata come

<lista di:<espressione>

che significa: <espressione>[,<espressione>]*

cioè una lista come:

3,4 oppure
3,4,4 oppure
3,4,5,6,7,8

La lista può essere costituita anche di un solo elemento; nel caso che essa ne comprenda più di uno, tutti quelli successivi al primo devono essere preceduti da una virgola, in modo da permettere al BASIC di trattare separatamente ognuno degli elementi.

I numeri possono essere rappresentati in vari modi:

a. notazione convenzionale:
   numeri reali privi della parte esponenziale
b. notazione scientifica:
   numeri formati da una mantissa e da un esponente come

   2E4 (cioè 2 × 10 elevato alla quarta)

l'esponente può essere sia positivo che negativo.

c. numeri in base:
   numeri espressi in forma binaria o esadecimale (vedere l'Appendice II):

forma Decimale (condizione di default) 100
forma Esadecimale &64 oppure &H64 . . . . (la H è opzionale)
forma Binaria &X1100100 . . . . (la X è obbligatoria)

3.5 — Un po' di pratica. Introduzione alla PRINT

In questa Sezione si trovano alcuni esempi che servono a far pratica con la terminologia utilizzata. Il comando PRINT è particolarmente adatto ad illustrare la maggior parte delle caratteristiche di questa terminologia. Per Comando si intende un'istruzione BASIC che può essere eseguita sia in modo diretto che in modo programma. Una Funzione richiede la presenza di un comando che la richiami, ad esempio:

```
PRINT FRE("")
```

Se volete ottenere dal computer risposte sensate dovete fornirgli tre indicazioni:

1. Dove far apparire la risposta. Su schermo, su stampante, o da qualche altra parte.

2. Quali sono i dati su cui lavorare.

3. Come lavorare sui dati.

La PRINT indica al computer su quale <canale> di output inviare la risposta. Ogni <canale> è identificato da un numero compreso fra 0 e 9 che assume il seguente significato:

0..7 sono canali testo collegati con le finestre video precedentemente aperte con il comando WINDOW.

8 è il canale stampante e può essere utilizzato solo se vi si trova connessa una stampante con interfaccia di tipo Centronics.

9 è il canale di output verso registratore e deve essere stato precedentemente aperto nel corso del programma.

Una forma semplice del comando PRINT, senza far ricorso alla clausola USING, può essere del tipo:

PRINT [<canale>][<lista di:<elementi da stampare>]

Le parentesi quadrate indicano che non è necessario dichiarare il canale, ne è necessario stampare qualcosa. Non specificando il canale il CPC 464 agisce come se fosse stato indicato il canale #0, cioé lo schermo a partire dalla posizione attuale del cursore. Provate l'esempio che segue, ricordandovi di premere [ENTER] per fare in modo che il comando venga accettato ed eseguito:

```
PRINT "SALVE"
```

Il computer risponderà:

```
SALVE
```

Notate che i doppi apici non compaiono assieme alla parola SALVE, perchè essi in BASIC indicano l'inizio e la fine di ciò che deve essere inviato su schermo.

Ora scrivete:

```
PRINT #0,"SALVE"
```

e il risultato è lo stesso.

Ma se scrivete:

```
PRINT #4,"SALVE"
```

la parola SALVE comparirà in alto a sinistra sullo schermo, perchè questa è la prima posizione libera del canale video #4, che per default copre l'intero schermo in quanto non precedentemente definito altrimenti con un comando WINDOW. Come per il #4, il punto di inizio di ogni canale video è l'angolo in alto a sinistra, da cui partono inizialmente tutti gli output.

La capacità del BASIC AMSTRAD di gestire più finestre video risulta molto utile per risolvere semplicemente anche i casi più complessi di organizzazione dello schermo.

Il BASIC visualizza qualsiasi cosa venga inserita fra doppi apici, anche parole riservate, senza entrare nel merito. Ad esempio:

```
PRINT "4*4"
```

darà come risultato:

```
4*4
```

Per indicare al BASIC di eseguire la moltiplicazione e di mostrarne il risultato è sufficiente togliere i doppi apici:

```
PRINT 4*4
```

produce infatti come risultato 16.

Notate che il numero 16 appare spostato di una colonna rispetto al margine sinistro dello schermo in quanto il BASIC riserva questo spazio per il segno, che per default compare solo se negativo.

La <lista di:<elementi da stampare> che segue il comando PRINT può essere composta da numeri, variabili e espressioni alfanumeriche (cioè da una variabile alfanumerica predefinita, o da qualsiasi altra cosa compresa fra doppi apici).

La clausola USING permette di definire particolari formati di output e serve quando sono necessari allineamenti con scarto di parti frazionarie o resti non desiderati.

3.6 — La clausola PRINT USING e le ZONE di stampa

All'accensione, il BASIC assume l'ampiezza di ogni ZONA di stampa pari a 13 colonne. Una virgola avrà come effetto di visualizzare il carattere seguente all'inizio della successiva ZONA di stampa. Richiedendo una tabulazione che eccede i limiti dello schermo, il BASIC stamperà dall'inizio della linea successiva.

Non specificando alcun tipo di USING, il BASIC porrà uno spazio prima dei numeri positivi e il segno meno prima dei negativi.

Il BASIC AMSTRAD non interpreta il tasto [TAB] come un vero e proprio tabulatore, in quanto non esiste uno standard per il significato di questo tasto nei vari dialetti del BASIC. Premendo il tasto [TAB] si ottiene la stampa di una freccia verso destra (la stessa che premendo [CTRL] e I assieme), senza alcun altro significato.

3.7 — PRINT TAB (<espressione intera>) (<lista di:<elementi>)

L'effetto di questo comando può essere facilmente illustrato con un esempio:

```
10  MODE 2: INK 1,0: INK 0,9
20  FOR N=1 TO 5
20  ZONE 40
30  PRINT TAB(N*4)"WOW",N
40  NEXT
```

Questo programma mostra sia l'effetto della virgola insieme al comando ZONE che il funzionamento del TAB. Modificate la riga 10 in questo modo:

10 FOR N=-5 TO 5

L'istruzione TAB fa in modo che la PRINT inizi a stampare da una posizione spostata verso destra del numero di colonne specificato dalla <espressione intera>.

La forma PRINT USING viene utilizzata per definire il formato dei risultati di quelle operazioni che produrrebbero cifre decimali non desiderate. È un concetto abbastanza complesso che può essere più facilmente compreso seguendo alcuni esempi pratici.

La forma del comando:

PRINT[#<canale>,][<listadi: <elementi>][<clausolaUSING>][<separatore>]

non è certamente delle più semplici, soprattutto perchè la <clausola USING> si suddivide ancora in:

USING <stringa>;[<lista>]

dove la <lista> si divide ulteriormente in:

<espressione>[<separatore><espressione>]*

Provate l'esempio seguente:

PRINT 123.456, USING "###.##";4567.896

Il risultato

123.456 %4567.90

illustra alcuni concetti fondamentali. Primo, che la clausola USING influenza unicamente gli <elementi> che la seguono. Secondo, che lo USING definisce il numero di posizioni sulle quali deve essere stampato l'elemento che segue e che, se le cifre a sinistra della virgola decimale sono superiori alle posizioni ammesse, l'elemento viene preceduto da un % per indicare un overflow. Inoltre, la virgola che segue 123.456 provoca la visualizzazione del secondo elemento all'inizio della successiva zona di stampa. Se si fosse trattato di un punto e virgola, il numero sarebbe stato stampato immediatamente alla destra del precedente, separato da uno spazio. I numeri stampati sulla medesima linea, per ovvie ragioni, vengono sempre separati da uno spazio, alla sinistra del segno.

Infine, notate come l'espressione venga arrotondata e non semplicemente troncata alla prima cifra non compresa nel formato definito dalla USING.

Ricordando che, di norma, i numeri negativi vengono preceduti dal segno meno e quelli positivi da uno spazio, è possibile tramite la USING modificare questa

situazione. Infatti, inserendo il segno + all'inizio o alla fine della dichiarazione del formato, si otterrà sempre la stampa del segno prima o dopo ogni numero. Alcuni esempi:

```
PRINT USING "+####.##";1234.599
+1234.60

PRINT USING "######+";—1234.599
1234.60—
```

CAPITOLO 4

VARIABILI, OPERATORI E DATI

La gestione delle informazioni in un programma BASIC.
Argomenti trattati in questo capitolo:
- Gestione
- Tipi di variabili: reali, intere e alfanumeriche
- Operatori, espressioni logiche
- Vettori e matrici
- I dati e il comando DATA

4.1 — Le parole riservate

Il comando LIST converte tutte le parole chiave del BASIC, anche se inserite in minuscolo, in lettere maiuscole. Naturalmente, eventuali parole scritte in modo errato restano in minuscolo in modo da facilitare la lettura e la correzione del listato.

Il BASIC AMSTRAD permette di includere parole chiave nei nomi delle variabili, dove per variabile si intende un nome assegnato ad un elemento specifico. Può trattarsi anche di una sola lettera (il primo carattere del nome di una variabile non può essere numerico), anche se, per facilitare la lettura e la correzione dei programmi è opportuno utilizzare nomi di variabili che hanno a che vedere con quanto accade durante l'esecuzione del programma. Ad esempio:

```
RISPOSTA = 4*4: PRINT RISPOSTA
```

Ogni variabile può avere un nome lungo fino a 40 caratteri (il primo deve essere una lettera), e tutti e 40 sono significativi. Il nome assegnato a una variabile non può contenere spazi, il primo dei quali provocherebbe l'arresto dell'esecuzione e la visualizzazione del messaggio:

```
Syntax error
```

Questo indica che è stata incontrata una sequenza illegale di caratteri (vedere Appendice VIII). Se proprio volete utilizzare nomi composti da più parole usate il punto invece dello spazio come separatore. Sono ammesse tutte le normali forme di variabili con indice, tenendo presente che ogni vettore deve essere dimensionato.

4.2 — Abbreviazioni

È possibile abbreviare la parola chiave PRINT con il punto interrogativo [?]: il BASIC la accetterà ugualmente. Notate che la forma abbreviata [?] non deve essere delimitata da uno spazio come invece avviene per la forma estesa PRINT. Inserite la riga di programma:

```
10?4*4
```

e date RUN

La risposta sarà ovviamente 16, ma chiedendo il listato vedrete comparire:

```
10 PRINT 4*4
```

Il BASIC ha inserito uno spazio prima di espandere il punto interrogativo. Nei comandi PRINT che utilizzano i doppi apici, come:

```
10?"SALVE"
```

è possibile eliminare l'ultimo doppio apice presente sulla riga, come mostrato anche dalla sequenza [CTRL][ENTER] utilizzata per leggere un programma da cassetta ed eseguirlo, che mostra sullo schermo la scritta RUN". Lo stesso può essere fatto all'interno delle righe di programma, ma non si ratta di una buona abitudine, soprattutto in previsione di dover tornare sulla riga per correzioni o aggiunte.

4.3 — Righe multi-istruzione e priorità matematica

Su di una singola riga di BASIC si possono eseguire più operazioni, tante quante se ne vuole, senza però superare il limite massimo di lunghezza della riga fissato in 255 caratteri. Le istruzioni presenti sulla stessa riga devono essere separate dai due punti [:].

È essenziale, al fine di evitare errori nelle operazioni matematiche, compren-

dere l'ordine in cui i vari operatori matematici vengono riconosciuti ed eseguiti dal BASIC. Essi sono:

↑ Esponenziale (elevamento a potenza).

— Meno unario (il segno utilizzato per indicare che un numero è negativo).

* Moltiplicazione.

/ Divisione.

\ Divisione intera (dal risultato viene scartata la parte decimale).

+ Addizione.

— Sottrazione.

La priorità rispetta l'ordine in cui gli operatori sono elencati.

Qualsiasi espressione inserita fra parentesi ( ) viene calcolata con priorità massima e se l'espressione consiste di più operazioni, queste verranno svolte secondo l'ordine sopra indicato, comprendendo anche altre eventuali parentesi. Ad ogni parentesi aperta deve corrispondere una parentesi chiusa, pena la segnalazione di un Syntax error.

4.4 — Un passo avanti

Fino a questo momento abbiamo praticamente usato il CPC 464 come una calcolatrice, pur avendo iniziato a comprenderne i segreti quando ci siamo occupati della PRINT nel Capitolo 3. Siamo quindi pronti ad occuparci dei vari comandi del BASIC, il cui elenco alfabetico con relativa descrizione si trova al Capitolo 8.

Molte istruzioni del BASIC conservano il significato letterale inglese, risultando perciò facilmente comprensibili a chi possiede le più elementari nozioni di questa lingua. Per fare solo gli esempi più ovvii, GOTO 50 vuol dire null'altro che 'vai alla riga 50' e END sta esattamente per 'fine'.

In modo diretto potete eseguire una serie di comandi separati dai due punti dopo la pressione del tasto [ENTER] senza bisogno del RUN. Ricordate però che per poter rieseguire la riga in questione sarà necessario ricorrere alla funzione di COPY.

4.5 — Operatori logici

Il BASIC utilizza in modo intensivo la capacità di un computer di eseguire ad alta velocità semplici operazioni ripetitive. Molti comandi sono disponibili per

permettere la generazione di questi processi ripetitivi (iterazione): comandi che iniziano, continuano e terminano un'iterazione al verificarsi di alcune condizioni predeterminate.

Per determinare la relazione esistente tra due dati si ricorre all'uso degli operatori relazionali, che permettono di confrontare variabili con variabili o variabili con costanti. Gli operatori relazionali sono:

<	minore di
<=	minore o uguale a
=	uguale a
>	maggiore di
>=	maggiore o uguale a
< >	diverso da

Ecco ora un breve esempio che illustra l'utilizzo degli operatori relazionali. Accendete il vostro CPC 464, se l'avete già fatto resettatelo premendo [CTRL][SHIFT][ESC] simultaneamente, e inserite il seguente programma:

```
10 INPUT "Qual'è il tuo stipendio"; STIPENDIO
20 IF STIPENDIO < 25000 THEN GOTO 30 ELSE 40
30 PRINT "CHIEDI UN AUMENTO": END
40 PRINT "COMPRATI UNA NUOVA AUTO"
```

Per eseguire questo semplice programma scrivete RUN seguito da [ENTER]. Il computer vi chiederà:

Qual'è il tuo stipendio?

(Notate come il computer inserisca automaticamente il punto interrogativo quando attende una dato come risposta). Rispondete alla domanda utilizzando solo numeri, non lettere o altro, e battete [ENTER] perchè la vostra risposta venga accettata.

Quando riappare il Ready, inserite la riga 5:

```
5  CLS
```

Rieseguite il programma per pulire lo schermo. Se la vostra prima risposta era stata inferiore a 25000, ora provate con un numero maggiore per vedere la differenza. Indi aggiungete al programma originale la riga 50:

```
5  CLS
10 INPUT "Qual'è il tuo stipendio"; STIPENDIO
20 IF STIPENDIO < 25000 THEN GOTO 30 ELSE 40
30 PRINT "CHIEDI UN AUMENTO":END
40 PRINT "COMPRATI UNA NUOVA AUTO"
50 IF STIPENDIO >75000 THEN PRINT "e cerca un buon contabile"
```

Il comando END della riga 30 termina l'esecuzione del programma e riporta al BASIC. Non essendoci alcun END alla riga 40, il programma procede per controllare se STIPENDIO è maggiore di 75000, nel qual caso risponde per le rime.

Andiamo avanti aggiungendo una nuova riga, la 60:

```
5 CLS
10 INPUT "Qual'è il tuo stipendio";STIPENDIO
20 IF STIPENDIO < 25000 THEN GOTO 30 ELSE 40
30 PRINT "CHIEDI UN AUMENTO":END
40 PRINT "COMPRATI UNA NUOVA AUTO"
50 IF STIPENDIO >75000 THEN PRINT "e cerca un buon contabile"
60 IF STIPENDIO >50000 THEN PRINT "e prestami un deca"
run
```

Notate che gli operatori relazionali (> e <) rendono inutile lo spazio prima e dopo il numero. Lanciando il programma e rispondendo con 55000, vedrete che il BASIC ignora la riga 50 (come se questa non esistesse) ed esegue invece le operazioni della riga 60.

A questo punto, create un programma tutto vostro utilizzando gli elementi del BASIC fin qui illustrati. Notate il modo in cui il programma è cresciuto; la maggior parte dei programmi si evolve in tal modo, introducendo forse il più importante concetto della programmazione in BASIC ......

4.6 — Evoluzione: l'origine dei programmi

La più interessante caratteristica del BASIC è il modo in cui questo permette di costruire programmi: una riga dopo l'altra. I puristi diranno che questa caratteristica porta all'utilizzo di pessime tecniche di programmazione, con i programmi che subiscono modifiche e miglioramenti all'arrivo di ogni nuova idea; le persone più pratiche possono considerare questo il miglior modo per ottenere buoni risultati, controllabili ad ogni stadio dello sviluppo.

Prendete nuovamente il programma d'esempio e aggiungete la riga 70, il cui compito è quello di riportare il programma all'inizio dopo aver sospeso l'esecuzione tanto quanto basta perchè possiate leggere la risposta sullo schermo:

```
5  CLS
10 INPUT "Qual'è il tuo stipendio";STIPENDIO
20 IF STIPENDIO < 25000 THEN GOTO 30 ELSE 40
30 PRINT "CHIEDI UN AUMENTO":END
40 PRINT "COMPRATI UNA NUOVA AUTO"
50 IF STIPENDIO >75000 THEN PRINT "e cerca un buon contabile"
60 IF STIPENDIO >50000 THEN PRINT "e prestami un deca"
70 for n=1 to 900: next n: goto 5
```

La nuova riga, e anche le altre che sono state aggiunte, sono state inserite in minuscolo per ricordarvi quanto detto all'inizio di questo capitolo. Per interrompere il programma premete due volte [ESC], poi chiedete il LIST e vedrete che il computer converte tutte le parole chiave in lettere maiuscole, ma lascia la variabile 'n' in minuscolo.

La riga 70 introduce un ciclo di ritardo in quanto il computer incrementa 'n' da 1 a 900 prima di eseguire l'istruzione successiva, cioè GOTO 5. Quindi il programma si ripete all'infinito. L'unica via d'uscita consiste nel tasto [ESC]; premendolo una volta si sospende l'esecuzione del programma, premendolo nuovamente si torna al modo diretto, senza però perdere il contenuto della memoria.

In effetti, a meno che non premiate [ESC] mentre il computer sta eseguendo il ciclo di ritardo della riga 70, il ritorno al modo diretto si otterrà a seguito della prima pressione del tasto [ESC] in quanto il CPC 464 non esegue alcunchè mentre attende un input. Premendo [ESC] mentre il computer è in attesa dell'input otterrete la segnalazione:

```
Break in 10
```

Interrompendo l'esecuzione durante il ciclo di ritardo, dopo la seconda pressione del tasto [ESC], verrà visualizzato il messaggio:

```
Break in 70
```

Premendo [ESC] una sola volta nel ciclo della riga 70, si sospende l'esecuzione, ma questa può essere ripresa semplicemente premendo un tasto qualsiasi, tranne [ESC]. Anche dopo aver interrotto l'esecuzione questa può essere ripresa dal punto in cui si era fermata, tramite il comando:

```
CONT
```

Qualunque cosa facciate con il tasto [ESC], non perderete mai il contenuto della memoria, a meno che non diciate esplicitamente al computer di cancellarla utilizzando il comando NEW, o resettando completamente il sistema con la pressione simultanea dei tasti [CTRL][SHIFT][ESC].

Pur essendo chiaro che la cancellazione del programma in memoria può avvenire solamente se e quando veramente voluta, per prudenza salvate su casseta tutto quanto pensate di poter riutilizzare in futuro.

4.7 — Tipi di variabili

La base del calcolo è la variabile. Ricordate che ogni espressione matematica contenente una parte variabile avrà pure un risultato variabile.

Le variabili hanno tre attributi o caratteristiche: un nome, un tipo e una 'organizzazione'. Tutto quanto riguarda i nomi è già stato discusso in precedenza. Dal momento che la definizione del tipo di variabile è opzionale, utilizzando il formalismo noto potremo descrivere una variabile come:

<nome>[<indicatore del tipo>]

L'<indicatore del tipo> può essere:

% indica le variabili Intere, dove tutto ciò che si trova alla destra della virgola decimale viene scartato. Le variabili Intere occupano meno spazio in memoria di quelle Reali, perciò i programmi che non necessitano di calcoli in virgola mobile possono essere resi più veloci e compatti utilizzando variabili Intere. Il comando DEFINT definisce una variabile come intera nel campo che va da −32768 a +32767.

! indica le variabili Reali, dove sia la parte intera che quella decimale vengono trattate. Per default tutte le variabili sono Reali, perciò si devono definire le variabili come Reali solo se precedentemente è stato utilizzato il comando DEFINT. Le variabili Reali possono assumere qualunque valore nel campo che va da 2.9E−39 a 1.7E+38.

$ indica le variabili Alfanumeriche, o stringa, dove il contenuto può essere un misto di numeri, lettere ecc. In altre parole, una qualunque sequenza di caratteri inclusa fra doppi apici. Per esempio:

```
NAME$ = "BOB SMITH"
```

Utilizzando questa nuova conoscenza aggiungete la riga 6 al nostro esempio e modificate la 60:

```
5 CLS
6 INPUT "Come ti chiami";NOME$
10 INPUT "Qual'è il tuo stipendio";STIPENDIO
20 IF STIPENDIO < 25000 THEN GOTO 30 ELSE 40
30 PRINT "CHIEDI UN AUMENTO":END
40 PRINT "COMPRATI UNA NUOVA AUTO"
50 IF STIPENDIO >75000 THEN PRINT "e cerca un buon contabile"
60 IF STIPENDIO >50000 THEN PRINT "e prestami un deca ";NOME$
70 for n=1 to 900: next n: goto 5
run
```

Il punto e virgola posto alla fine di un comando INPUT O PRINT fa in modo che il computer non vada a capo dopo aver eseguito il comando, come altrimenti farebbe.

Possiamo ulteriormente ampliare il programma aggiungendo la riga 61:

```
61 OGNI.GIORNO=STIPENDIO/365:PRINT "guadagni L.";
OGNI.GIORNO; "al giorno"
70 FOR n=1 to 5000: NEXT n: GOTO 5
run
```

Notate che è stato aumentato il ritardo provocato dal ciclo della riga 70, essendoci ora un maggior numero di cose da leggere sullo schermo. Il risultato del calcolo del guadagno giornaliero può essere arrotondato ad un valore intero con l'inserimento della riga 62:

```
62 OGNI.GIORNO.INT%=OGNI.GIORNO:PRINT "oppure L.";
OGNI.GIORNO.INT%;"se non ti preoccupi troppo degli spiccioli"
run
```

Ricordatevi di inserire nel nome della variabile l'indicatore del tipo intero [%], perchè potrebbe esistere un'altra variabile Reale con lo stesso valore. Avrete sicuramente notato come vengano spezzate le righe di programma quando la loro lunghezza supera quella di una linea dello schermo. Questo accade in tutti e tre i Modi.

Utilizzate il Modo 2 per scrivere programmi lunghi, in quanto è più semplice leggere listati le cui righe non vengono continuamente spezzate sul margine.

Per selezionare il Modo 2, battete:

```
MODE 2
```

e per ottenere lo schermo in bianco e nero, che è più leggibile con il CTM640, inserite i seguenti comandi diretti:

```
INK 1,0
INK 0,13
BORDER 13
```

Ora chiedete il LIST.

4.8 — L'organizzazione dello schermo

Il nostro programma a questo punto deve essere rimesso un po' in ordine. La prima cosa che possiamo fare è rinumerare tutte le righe utilizzando il comando RENUM. In modo diretto, scrivete:

```
RENUM
```

e chiedete nuovamente il LIST:

```
10 CLS
20 INPUT "Come ti chiami";NOME$
30 INPUT "Qual'è il tuo stipendio";STIPENDIO
40 IF STIPENDIO < 25000 THEN GOTO 50 ELSE 60
50 PRINT "CHIEDI UN AUMENTO":END
60 PRINT "COMPRATI UNA NUOVA AUTO"
70 IF STIPENDIO >75000 THEN PRINT "e cerca un buon contabile"
80 IF STIPENDIO >50000 THEN PRINT "e prestami un deca ";NOME$
90 OGNI.GIORNO=STIPENDIO/365:PRINT "guadagni L.";
OGNI.GIORNO; "al giorno"
100 OGNI.GIORNO.INT%=OGNI.GIORNO:PRINT "oppure L.";
OGNI.GIORNO.INT%;"se non ti preoccupi troppo degli spiccioli"
110 FOR n=1 TO 5000: NEXT n: GOTO 10
```

Tutte le righe sono state rinumerate e le istruzioni di salto aggiornate secondo i nuovi numeri di riga.

Ora passiamo a fare un po' di pulizia sullo schermo, disabilatando, come prima operazione, il ciclo della riga 110 con l'inserimento di una REM: all'inizio. In questo modo il BASIC ignorerà la riga 110 e lascerà tutte le scritte sullo schermo. Battendo in seguito:

```
15 mode 1
```

si otterrà la selezione del Modo di organizzazione dello schermo, indipendentemente da ogni precedente scelta. Notate che, siccome il comando MODE provvede automaticamente ad eseguire una CLS, la riga 10 risulta adesso ridondante. Non preoccupatevi e lasciatela come è. Dando il RUN sullo schermo compariranno le righe seguenti:

```
Come ti chiami? Roberto
Qual'è il tuo stipendio? 77000
COMPRATI UNA NUOVA AUTO
e cerca un buon contabile
e prestami un deca Roberto
guadagni L. 210.958904 al giorno
oppure L. 211
se non ti preoccupi troppo degli spicci
oli
Ready
```

Il tutto non è molto elegante, specialmente la parola 'spiccioli' spezzata inopinatamente su due righe.

Aggiungete le due righe:

```
25 PRINT: PRINT
85 PRINT
```

e modificate la riga 100:

```
100 OGNI.GIORNO.INT%=OGNI.GIORNO:PRINT "oppure L."; O-
GNI.GIORNO.INT%;"se non ti preoccupi troppo": PRINT "degli spiccioli"
```

Date nuovamente il RUN e vedrete che le parole 'se non ti preoccupi troppo' compariranno sulla stessa riga di 'oppure L. ..', dato che c'è posto. Per spostare verso il basso il messaggio Ready inserite la riga 120:

```
120 ?:?:?:?:?
```

detto messaggio può essere evitato, impedendogli di comparire, con una riga del tipo: 120 goto 120

L'unico modo per interrompere l'esecuzione di quest'ultima versione del nostro programma consiste nel premere il tasto [ESC]. A questo punto il listato è:

```
10 CLS
20 INPUT "Come ti chiami";NOME$
25 PRINT: PRINT
30 INPUT "Qual'è il tuo stipendio";STIPENDIO
40 IF STIPENDIO < 25000 THEN GOTO 50 ELSE 60
50 PRINT "CHIEDI UN AUMENTO":END
60 PRINT "COMPRATI UNA NUOVA AUTO"
70 IF STIPENDIO >75000 THEN PRINT "e cerca un buon contabile"
80 IF STIPENDIO >50000 THEN PRINT "e prestami un deca";NOME$
85 PRINT
90 OGNI.GIORNO=STIPENDIO/365:PRINT "guadagni L.";
OGNI.GIORNO; "al giorno"
100 OGNI.GIORNO.INT%= OGNI.GIORNO:PRINT "oppure L.";
OGNI.GIORNO.INT%;"se non ti preoccupi troppo":
PRINT "degli spiccioli"
110 REM:FOR n=1 TO 5000: NEXT n: GOTO 10
120 GOTO 120
```

4.9 — Il comando LOCATE

Fino ad ora, quasi tutti i comandi sono stati scritti utilizzando quella che è la sintassi 'universale' del BASIC comprensibile dalla maggioranza dei computer dotati di un qualsiasi tipo di interprete BASIC. Il comando LOCATE è invece pro-

prio del dialetto dell'AMSTRAD, ma anche di altri, e permette di posizionare il cursore dovunque sullo schermo:

```
LOCATE 10,4
```

posiziona il cursore alla decima colonna sulla quarta linea dall'alto dello schermo. Utilizzandolo come comando diretto non è possibile vederne l'effetto, in quanto il Ready del Basic manda sempre il cursore a capo. Aggiungete al programma la riga:

```
16 LOCATE 10,4
run
```

la prima richiesta di input è stata spostata verso il basso e posizionata di conseguenza alla LOCATE. Fermate il programma con la duplice pressione del tasto [ESC] e inserite la riga:

```
26 LOCATE 10,4
```

Ridate RUN e la seconda domanda si sovrapporrà alla prima. Ora potete posizionare il testo esattamente dove volete che appaia sullo schermo, utilizzando per comodità la griglia delle coordinate di schermo di cui all'Appendice VI.

Se volete che tutte le domande ed i commenti appaiano sulla medesima linea è opportuno che facciate precedere il LOCATE da un CLS, in modo da evitare che si crei confusione con quanto rimasto del precedente inserimento.

Le coordinate indicate nel comando LOCATE possono essere contenute in variabili, definite all'interno del programma. Abbiamo fatto tutto quanto era possibile con il programma 'stipendio', d'ora in avanti i nostri esempi si baseranno su argomenti differenti. Se volete salvare quanto avete costruito finora, potete farlo utilizzando una cassetta e i comandi descritti nel Capitolo 2.

4.10 IF...THEN

In inglese significa 'se.....allora...' e in BASIC il suo utilizzo corrisponde letteralmente a tale significato. Una delle tante forme di questo comando è: IF <espressione logica> THEN GOTO <numero di riga>.

La parte IF controlla se il risultato della <espressione logica> è vero, nel qual caso tutti i comandi che seguono la THEN vengono eseguiti. Resettate il computer con la sequenza [CTRL][SHIFT][ESC] e inserite il breve programma che segue:

```
1 MODE 1
10 AMSTRAD=0
20 PRINT "Personal Computer AMSTRAD CPC 464"
```

```
30 AMSTRAD=AMSTRAD +1
40 IF AMSTRAD <10 GOTO 20
```

Date il RUN e vedrete che il comando PRINT della riga 20 viene ripetuto sino a quando la condizione posta alla riga 40 è soddisfatta. Con il termine 'variabile' intendiamo che, ad esempio, AMSTRAD cambia il suo valore ad ogni ciclo del programma.

Se volete vedere come si modifica il valore di AMSTRAD durante il corso del programma, potete inserire una nuova riga:

```
35 LOCATE 1,20:PRINT AMSTRAD:LOCATE 1,AMSTRAD
run
```

Rallentate l'esecuzione, se volete, inserendo un ciclo di ritardo:

```
36 for n=1 to 500:next
```

ora diamo un po' di colore con la riga:

```
34 BORDER AMSTRAD
```

Questa riga provoca il cambiamento del colore del bordo a seconda del valore della variabile AMSTRAD. A questo punto chiedete il LIST del programma:

```
1 MODE 1
10 AMSTRAD=0
20 PRINT "Personal Computer AMSTRAD CPC 464"
30 AMSTRAD = AMSTRAD + 1
34 BORDER AMSTRAD
35 LOCATE 1,20:PRINT AMSTRAD:LOCATE 1,AMSTRAD
36 FOR n=1 TO 500:NEXT
40 IF AMSTRAD <10 GOTO 20
run
```

Non c'è dubbio che vorrete vedere tutti i colori disponibili sul CPC 464: per ottenerli è sufficiente modificare la riga 40

```
40 IF AMSTRAD <26 GOTO 20
```

Date il RUN al programma e vedrete tutti i colori disponibili partendo dal più scuro e terminando con il bianco luminoso.

Potete inserire un messaggio alla riga 35 per indicare più esplicitamente il colore del bordo, ad esempio:

```
35 LOCATE 1,20:PRINT "Bordo";AMSTRAD:LOCATE 1,AMSTRAD
run
```

Sempre rimanendo nell'argomento bordi, quando il programma termina e ritorna il Ready, inserite

BORDER 14,6

e vedrete il colore del bordo alternarsi fra il blu pastello ed il rosso, cioè i colori, rispettivamente, di codice 14 e 6. Dovete attendere il prossimo capitolo per ottenere nuove informazioni circa l'utilizzo dei colori. Per terminare questa sotto-sezione, inserite:

ink 1,18,16

e poi

speed ink 1,5

Salvate il tutto per stupire in futuro i vostri amici.

4.11 — I vettori

Alcuni programmi richiedono un gran numero di variabili per memorizzare i dati. Fortunatamente, il BASIC vi viene incontro permettendovi di gestire i vettori.

Cosa è un vettore ? Potreste chiedere. Fondamentalmente si tratta di un gruppo di variabili tutte identificate dallo stesso nome.

Il modo migliore per illustrarvi il significato e l'utilizzo dei vettori è certamente un esempio pratico. Immaginate un programma che simuli un qualunque gioco con le carte e che richieda anche la distribuzione casuale delle mani. Ovviamente, la stessa carta non può essere distribuita due volte, è quindi necessario ricordarsi di ogni carta estratta dal mazzo. Un modo semplice per risolvere questo problema consiste nell'associare una variabile ad ogni carta e assegnare alla variabile un valore differente a seconda che la carta corrispondente sia stata distribuita o meno (supponiamo 1 per indicare che è stata estratta e 0 quando si trova ancora nel mazzo).

Evidentemente, questo procedimento richiede 52 variabili separate e implica che voi ricordiate la variabile associata ad ogni carta: ecco un tipico caso in cui i vettori risultano praticamente indispensabili.

Per prima cosa, dobbiamo dare al vettore un nome, diciamo MAZZO. Ora, per accedere ad un particolare elemento del vettore è sufficiente indicarne la posizione, o meglio l'indice. Così, ad esempio, al sei di cuori sarà associato il sesto elemento del vettore di nome MAZZO, cioè MAZZO(6), al dieci di cuori MAZZO(10) e così via. Chiaro?

Sfortunatamente, non potete permettervi di non indicare al computer quanto spazio deve riservare in memoria per contenere tutti gli elementi del vettore. Il comando DIM serve proprio a questo, a definire cioè il numero di elementi che compongono un vettore, o meglio la sua dimensione. Perciò nel nostro caso sarà necessario il comando BASIC:

DIM MAZZO(52) o meglio DIM MAZZO(51) in quanto la numerazione degli elementi del vettore parte sempre da 0.

Ora possiamo scrivere la parte principale di un sottoprogramma per la distribuzione delle ccc carte:

```
10 DIM MAZZO(52)
20 FOR X = 1 TO 52
30 LET MAZZO(X) =0
40 NEXT X
50 .........
60 ......
1000 CARTA = RND(52) +1
1010 IF MAZZO(CARTA) = 1 THEN GOTO 1000
1020 MAZZO(CARTA) = 1
1030 RETURN
```

Notate che il comando di dimensionamento del vettore si trova nella prima riga del programma, questo perchè un vettore può essere dimensionato una sola volta nel corso dello stesso programma.

Il ciclo che va dalla riga 20 alla 40 pone a 0 tutti gli elementi del vettore. Il sottoprogramma che inizia alla riga 1000 sceglie casualmente una carta e controlla che non sia già stata estratta. Se sì, ne sceglie un'altra e così fino a quando non ne viene trovata una ancora non estratta. La routine modifica anche il valore dell'elemento appropriato del vettore per indicare che la carta corrispondente è stata distribuita, poi ritorna al programma principale.

È possibile utilizzare vettori multidimensionali. Questo si ottiene semplicemente aggiungendo ulteriori indici di riferimento ad ogni elemento del vettore. Per esempio, un vettore 10 × 10, meglio chiamarlo matrice, può essere dimensionato con il comando:

DIM VETTORE(10,10)

Questa tecnica è utile per dividere grandi insiemi di dati in sottoinsiemi più ristretti, semplificandone così la gestione. Nel nostro esempio, avremmo potuto dividere il MAZZO nei quattro semi utilizzando la forma:

DIM MAZZO (4,13)

Volendo trovare il quattro di picche, che sarebbe stato l'elemento 43 del vettore originale, dobbiamo semplicemente esaminare l'elemento (2,4), assumendo che le picche si trovino nella seconda riga della matrice.

I vettori possono essere utilizzati anche per la memorizzazione di dati alfanumerici. Un'applicazione in questo caso potrebbe consistere nella lista delle prenotazioni di un teatro o di un volo d'aereo.

4.12 — Il comando DATA

Questo comando, insieme con la READ, può essere utilizzato per inserire dati all'interno di un programma. La lista dei dati si trova su di una riga aperta con il comando DATA, gli elementi che la compongono devono essere separati da una virgola. La lista viene scandita sequenzialmente dal comando READ, che si occupa di assegnare i dati alle variabili. Ecco un breve esempio:

```
10 READ X,Y,Z
20 PRINT X;"+";Y;"+";Z;" = ";X+Y+Z
30 DATA 1,3,5
```

I dati possono essere sia numerici che alfanumerici. Se una riga non è sufficiente per contenere tutti i vostri dati, non preoccupatevi: apritene una nuova sempre con il comando DATA. La READ provoca la scansione dell'intero programma alla ricerca del successivo elemento di una lista DATA, senza preoccuparsi della posizione in cui questo si trova. Assicuratevi che ci siano dati a sufficienza per tutti i comandi READ presenti nel programma, altrimenti vi verrà segnalato un errore.

L'unico modo per interrompere l'input sequenziale dei dati consiste nell'uso della RESTORE, che rimanda il puntatore dei dati all'inizio del programma, permettendo, quando richiesto, anche la rilettura degli stessi dati. L'esempio chiarisce ogni cosa:

```
10 FOR C = 1 TO 4
20 READ X$
30 PRINT X$;" ";
40 NEXT
50 RESTORE
60 GOTO 10
70 DATA CIAO,COME,STAI,OGGI
```

Premete [ESC] per uscire dal programma.

Quantunque la riga DATA sia stata posta alla fine del programma, ribadiamo che la posizione è irrilevante.

Il comando DATA non è utilizzato solo per contenere informazioni da passare ad una PRINT, ma, per esempio, può servire anche per fornire dati ad un comando SOUND, come nell'esempio:

```
10 FOR n=1 TO 30
20 READ s
30 SOUND 1,s,40,5
40 NEXT n
50 DATA 100,90,100,110,120,110,100,0
60 DATA 130,120,110,0,120,110,100,0
70 DATA 100,90,100,110,120,110,100,0
80 DATA 130,0,100,0,120,150
```

Se non riuscite ad udire alcunchè, non date la colpa al programma ma al controllo del volume che si trova sulla destra del CPC 464.

Per concludere questa introduzione al BASIC, vi proponiamo un programma per giocare a Black Jack. Pur avendo fatto ricorso a molte delle possibilità del BASIC, potete sbizzarrirvi ad aggiungere effetti grafici e sonori, procedendo, come di norma in BASIC, un passo dopo l'altro fino ad ottenere il migliore dei risultati.

Scopo del gioco è ottenere un valore quanto più vicino a 21 sommando i valori delle singole carte distribuite dal banco. Vi battete contro il banco e fate attenzione a non 'sballare' superando 21. Usate il comando AUTO per la generazione automatica del numero di riga, dopo aver inserito manualmente la riga 1.

```
1 REM BLACK JACK
10 REM INIZIALIZZAZIONE
20 yc=2:cc=2
30 ASSI = 0
40 CASSI = 0
50 s=0
60 t=0
70 DIM SEME$(4)
80 SEME$(1) = "FIORI"
90 SEME$(2) = "CUORI"
100 SEME$(3) = "PICCHE"
110 SEME$(4) = "QUADRI"
120 CLS
130 DIM MAZZO(52)
```

```
140 FOR x=1 TO 52
150 MAZZO(X) = 0
160 NEXT x
170 REM DISTRIBUISCE DUE CARTE AD OGNI GIOCATORE
180 LOCATE 10,3
190 PRINT "TU";SPC(15)"BANCO"
200 LOCATE 3,5
210 GOSUB 740
220 s=s+f
230 IF F=11 THEN ASSI = ASSI + 1
240 LOCATE 3,6
250 GOSUB 740
260 s=s+f
270 IF F=11 THEN ASSI = ASSI + 1
280 LOCATE 24,5
290 GOSUB 740
300 t=t+f
310 IF F=11 THEN CASSI = CASSI + 1
320 LOCATE 24,6
330 GOSUB 740
340 t=t+f
350 IF F=11 THEN CASSI = CASSI + 1
360 REM OPZIONE TIRO (T) O STO (S)
370 x$=INKEY$:IF x$<>"S" AND x$<>"T" THEN 370
380 IF x$="S" THEN 560
390 LOCATE 3,yc+5
400 yc=yc+1
410 GOSUB 740
420 s=s+f
430 IF F=11 THEN ASSI = ASSI +1
440 REM CONTROLLO PUNTEGGIO E ASSI
450 IF s<22 THEN 370
460 IF ASSI =0 THEN 500
470 ASSI = ASSI -1
480 s=s-10
490 GOTO 450
500 LOCATE 12,19
510 PRINT "SBALLATO !!"
520 PRINT:PRINT "UN'ALTRA PARTITA (S/N)"
530 X$=INKEY$:IF X$< >"S" AND X$< >"N" THEN 530
540 IF X$="S"THEN RUN
550 END
560 IF t>16 THEN 700
```

```
570 cc=cc+1
580 LOCATE 24,cc+4
590 gosoub 740
600 t=t+f
610 IF F=11 THEN CASSI = CASSI -1
620 IF t<21 THEN 560
630 IF CASSI =0 THEN 670
640 CASSI = CASSI -1
650 t=t-10
660 GOTO 620
670 LOCATE 12,19
680 PRINT "HAI VINTO !!"
690 GOTO 520
700 LOCATE 2,19
710 IF t<s THEN 680
720 PRINT "IL BANCO VINCE"
730 GOTO 520
740 REM DISTRIBUZIONE CARTE
750 LET CARTA = INT(RND(1)*52+1)
760 IF MAZZO(CARTA) = 1 THEN 750
770 MAZZO(CARTA) = 1
780 F = CARTA-13*INT(CARTA/13)
790 IF f=0 THEN f=13
800 IF f=1 OR f>10 THEN 850
810 PRINT F;" DI ";
820 IF f>0 THEN f=10
830 PRINT SEME$(INT((CARTA-1)/13) +1)
840 RETURN
850 IF F=11 THEN PRINT "JACK DI ";
860 IF F=12 THEN PRINT "DONNA DI ";
870 IF F=13 THEN PRINT "RE DI ";
880 IF f<>1 THEN GOTO 80
890 f=11
900 PRINT "ASSO DI ";
910 GOTO 830
```

Battete T per ‘tirare’ una carta il cui valore va ad aggiungersi a quello che già possedete, oppure S per ‘stare’ e passare la mano al banco. Non sosteniamo di aver prodotto l’ultimo grido in fatto di giochi di carte, ma vi abbiamo sicuramente fornito un’ottima base di partenza sulla quale costruire gli effetti grafici e sonori.

4.13 — Espressioni Logiche

Una delle differenze fondamentali tra una calcolatrice ed un computer consiste nella capacità di quest'ultimo di eseguire operazioni logiche come quelle condizionali di tipo IF..THEN. Gli operatori logici agiscono in base al valore binario dei campi che contengono gli operandi. Essendo notoriamente difficile descrivere in modo semplice argomenti di logica, utilizzeremo alcune definizioni coincise.

Un'espressione logica è normalmente composta di due parti, chiamate argomenti:

<argomento>[<operatore logico><argomento>]

dove

<argomento> può essere: NOT <argomento>
oppure <espressione numerica>
oppure <espressione relazionale>
oppure (<espressione logica>)

Entrambi gli argomenti di una operazione logica sono considerati come interi, producendo un Error 6 nel caso che vadano al di fuori dei limiti ammessi per gli interi.

Gli operatori logici sono i seguenti in ordine di priorità:

AND pone un bit a 1 nel risultato solo se i bit corrispondenti degli argomenti sono entrambi a 1.

OR pone un bit a 0 nel risultato solo se i bit corrispondenti degli argomenti sono entrambi a 0.

XOR pone un bit a 0 nel risultato solo se i bit corrispondenti degli argomenti sono uguali.

Alcuni esempi:

PRINT 10 AND 10

Fornisce come risultato 10

PRINT 10 AND 12

Fornisce come risultato 8

PRINT 10 AND 1000

Fornisce ancora 8

Questo perchè la rappresentazione binaria dei numeri 10 e 1000 è, rispettivamente:

```
      1010
1111101000
```

Con l'operatore AND si controlla la corrispondenza bit a bit e si costruisce il risultato ponendo 1 quando i bit corrispondenti in entrambe le file sono a 1 e 0 in tutti gli altri casi. Pertanto:

```
0000001000
```

Tutto questo significa che l'operatore logico AND viene usato per controllare la presenza simultanea di due condizioni. Un esempio chiarificatore:

```
10 CLS:INPUT "Il numero del giorno";giorno
20 INPUT "Il numero del mese";mese
30 IF giorno=25 AND mese=12 GOTO 50
40 GOTO 10
50 PRINT "Buon Natale"
```

Anche la OR lavora sui bit ponendo 1 nel risultato a meno che i bit corrispondenti degli argomenti non siano entrambi a 0.

Pertanto:

```
PRINT 1000 OR 10
```

darà come risultato

```
1002
```

che in notazione binaria è

```
1111101010
```

Esempio pratico:

```
10 CLS
20 INPUT "Il numero del mese";mese
30 IF mese=12 OR mese=1 OR mese=2 GOTO 50
40 GOTO 10
50 PRINT "È inverno, brrr.... che freddo....."
```

oppure

```
10 CLS
20 INPUT "Il numero del mese";mese
30 IF NOT(mese=6 OR mese=7 OR mese=8) GOTO 50
40 GOTO 10
50 PRINT "Peccato non sia estate"
```

Per concludere, tenete presente che è ammesso combinare tra loro più condizioni logiche per sofisticare al massimo le procedure di controllo:

```
10 CLS:INPUT "Il numero del giorno";giorno
20 INPUT "Il numero del mese";mese
30 IF NOT (mese=12 OR mese=1) AND giorno=29 GOTO 50
40 GOTO 10
50 PRINT "Non siamo in Dicembre nè in Gennaio, ma potrebbe essere un
anno bisestile"
```

I valori logici VERO e FALSO sono rispettivamente −1 e 0. Nel primo caso tutti i bit sono posti a 1, mentre nel secondo sono tutti a 0. Per avere una riprova di questo aggiungete al programma le righe 60 e 70:

```
60 PRINT NOT(mese=12 OR mese=1)
70 PRINT (mese=12 OR mese=1)
```

Al RUN, inserite 29 per il giorno e, ad esempio, 5 per il mese. Otterrete così la risposta della riga 50 e la visualizzazione dei valori attuali delle espressioni logiche.

Infine, la XOR (OR esclusivo) fornisce come risultato VERO solo quando i due argomenti sono diversi. Segue ora una tabella, nota come tavola della verità, dalla quale si desume facilmente come agiscono gli operatori logici sui bit:

ARGOMENTO A	:	1	0	1	0
ARGOMENTO B	:	0	1	1	0
A AND B	:	0	0	1	0
A OR B	:	1	1	1	0
A XOR B	:	1	1	0	0
	:				

CAPITOLO 5

GUIDA ALL'UTILIZZO DEL COLORE E DELLA GRAFICA

Argomenti trattati in questo capitolo:
- I Modi di organizzazione dello schermo
- I colori
- INK, PAPER e PEN
- Come disegnare sullo schermo
- Le finestre (WINDOW)

5.1 — Caratteristiche peculiari del CPC 464

Il BASIC AMSTRAD è sostanzialmente conforme allo standar industriale, per cui le differenze con altri 'dialetti' del BASIC sono minime, soprattutto per quanto riguarda le operazioni puramente aritmetiche; i comandi del BASIC dedicati alla gestione della grafica sono invece appositamente strutturati in base alle caratteristiche hardware del CPC 464 e richiedono un esame più approfondito.

5.1.1 — La selezione dei colori

Nel prosieguo di questo manuale il 'nero' (cioè l'assenza di colore) è considerato a tutti gli effetti come un colore.

Per il bordo (BORDER) è possibile selezionare una coppia qualsiasi di colori, indipendentemente dal Modo (MODE) in cui ci si trova; anche cambiando Modo il colore del bordo non viene interessato. Assegnando al bordo due colori, gli stessi saranno visualizzati in alternanza fra loro, producendo un effetto di flashing.

Il numero di colori contemporaneamente visualizzabili ed il numero di inchiostri (INK) utilizzabili dipendono dal Modo selezionato. Ad ogni INK è possibile

associare una coppia di colori in modo da ottenere il lampeggio. Il colore di fondo dei caratteri (PAPER), quello con cui vengono stampati (PEN) e quello della Penna Grafica possono essere associati ad uno qualsiasi degli inchiostri disponibili.

5.1.2 — L'opzione di trasparenza e la relazione tra INK, PEN e PAPER

La stampa della matrice di un carattere sullo schermo comporta normalmente l'utilizzo di due inchiostri: quello della PEN associato al colore con cui vengono accesi i bit posti ad 1 e quello del PAPER per i bit a 0.
NB. Il parametro del comando PAPER indica un codice e si riferisce all'INK associato a questo codice, e NON al numero del colore. Allo stesso modo, il parametro del comando PEN si riferisce all'INK dichiarato per la PEN di codice specificato, e NON al numero del colore di cui all'Appendice VI.

I codici degli INK associati per default al PAPER e alla PEN sono rispettivamente 0 e 1. Per attribuire all'INK di codice 0 il colore verde (numero 9), digitate:

```
INK 0,9
```

Allo stesso modo, per attribuire all'INK di codice 1 il colore nero (numero 0) digitate:

```
INK 1,0
```

Eseguendo il comando INK 0,0, assegnando cioè al PAPER lo stesso colore della PEN, si porta al nero l'intero schermo, rendendo praticamente invisibile ogni carattere scritto.

La stampa del testo può essere definita sia trasparente che opaca; questi effetti vengono ottenuti tramite una serie di caratteri di controllo, che costituiscono utili estensioni ai comandi grafici del BASIC. Con l'opzione di trasparenza è possibile sia ignorare il colore di fondo, e quindi sovrascrivere la pagina grafica, oppure sovrapporre totalmente i caratteri a quanto già presente sullo schermo. Il breve programma che segue mostra gli effetti ottenibili con l'utilizzo dei diversi modi di stampa:

```
[CTRL][SHIFT][ESC]
 10 MODE 1
 20 INK 2,19
 30 DRAW 200,200,2
 40 LOCATE 1,21
 50 PRINT "Stampa normale"
 60 PRINT CHR$(22)+CHR$(1)
 70 ORIGIN 0,0
```

```
 80 DRAW 500,200,2
 90 LOCATE 12,18
100 PRINT"Opzione di trasparenza"
110 PRINT CHR$(22)+CHR$(0)
120 LOCATE 22,15
130 PRINT"Stampa normale"
```

Il comando DRAW che si trova alla riga 30 viene eseguito prima di selezionare il modo trasparente, mentre quello della riga 80 è interpretato dopo l'esecuzione del comando di selezione del modo trasparente presente alla riga 60. Notate come i punti di incrocio cambino colore e come (in modo trasparente) il colore di riempimento della matrice del carattere risulti totalmente sovrascritto.

Scambiate fra loro le righe in cui si trovano i comandi di abilitazione del modo trasparente (riga 60) e quelli di disabilitazione (riga 80) e osservate gli effetti di questa operazione su quanto stampato sullo schermo. Un elenco completo di tutti questi comandi addizionali si trova nell'Appendice VI.

5.2 — Modi di organizzazione dello schermo

Sono disponibili tre Modi di organizzazione dello schermo:

a) Normale (MODE 1)

Modo 1: 40 colonne × 25 righe, 4 INK per il testo
320 × 200 punti, ognuno singolarmente indirizzabile con 4 colori

b) Modo multicolore (MODE 0)

20 colonne × 25 righe, 16 INK per il testo
160 × 200 punti, ognuno singolarmente indirizzabile con 16 colori

c) Modo di Alta Risoluzione (MODE 2)

80 colonne × 25 righe, 2 INK per il testo
640 × 200 punti, ognuno singolarmente indirizzabile con 2 colori.

Come si può notare, la differenza fra i tre Modi consiste nel numero di 'elementi' in cui è divisa ogni linea dello schermo (colonne o punti).

Tramite il BASIC è possibile utilizzare un solo Modo alla volta. Il passaggio da un Modo all'altro provoca la pulizia dello schermo e la chiusura di ogni finestra, sia di testo che grafica, eventualmente aperta; la stessa cosa che accade ese-

guendo un comando CLG o CLS. Il programma e i dati presenti in memoria non vengono comunque modificati dall'esecuzione del comando MODE. Tale comando, utilizzato appunto per il cambiamento del Modo, può essere inserito nel programma BASIC o eseguito direttamente.

5.2.1 — MODE 0, multicolore

Possono essere visualizzati contemporaneamente 16 dei 27 colori disponibili e ogni singolo elemento dello schermo (pixel) può essere colorato con uno di essi. Lo schermo consiste di 160 pixel per ogni riga orizzontale e 200 per ogni colonna verticale. Uno schema di questa 'griglia' si trova nell'Appendice VI.

Nel MODE 0 ognuna delle 25 righe di testo contiene 20 caratteri.

5.2.2 — MODE 1, il Modo standard

All'accensione del CPC 464 viene selezionato il MODE 1. Possono essere visualizzati contemporaneamente 4 dei 27 colori disponibili. Lo schermo consiste di 320 pixel per ogni riga orizzontale e 200 per ogni colonna verticale. Uno schema di questa 'griglia' si trova nell'Appendice VI.

Nel MODE 1 ognuna delle 25 righe di testo contiene 40 caratteri.

5.2.3 — MODE 2, Modo di Alta Risoluzione

Possono essere visualizzati simultaneamente 2 colori. Il MODE 2 viene utilizzato principalmente per la possibilità che ha di visualizzare 80 caratteri per ogni riga: questo rende più facile la stesura dei programmi in quanto sullo schermo viene listato un maggior numero di righe.

Il MODE 2 permette di indirizzare 640 pixel orizzontalmente e 200 verticalmente.

5.2.4 — Provate questo programma di esempio...

```
 5 REM Demo grafico
10 MODE 1
15 INK 2,0
16 INK 3,6:REM selezione del colore per il PLOT della riga 41
17 BORDER 1:REM blu scuro
20 CLG:REM pulizia della pagina grafica
30 b%=RND*5+1
```

```
 40 c%=RND*5+1
 50 ORIGIN 320,200
 60 FOR a=0 TO 1000 STEP PI/300
 70 x%=100*COS(a)
 80 MOVE x%,x%:REM spostamento del cursore grafico
 90 DRAW 200*COS(a/b%),200*SIN(a/b%),3
 91 IF INKEY$<>" " THEN 20
100 NEXT
110 GOTO 20
```

Lanciate il programma e premete un tasto qualunque per ottenere un nuovo disegno. Questo programma mostra alcune importanti capacità del software e dell'hardware del CPC 464: la 'scrittura' su schermo non provoca la perdita del sincronismo video e il software comprende comandi che permettono di ottenere effetti sofisticati con il minimo sforzo. Gli statement REM sono inclusi nel listato per commentarlo, ma non è necessario che voi li inseriate nel programma per farlo funzionare.

Salvate il programma su di una cassetta con un comando come:

```
SAVE "GRAFICA 5.2.4"
```

Il programma seguente disegna delle figure simmetriche:

```
new
 10 a$=INKEY$:REM Premere un tasto per iniziare
 20 IF a$=" " THEN 10
 30 CLS
 40 n=INT(RND*3):REM genera un numero casuale compreso tra 0 e 3
 50 IF m>2 THEN 40
 60 MODE m
 70 i1=RND*26:REM selezione del numero di colore
 80 i2=RND*26
 90 IF ABS(i1-i2)<5 THEN 70
100 INK 0,i1:INK 1,i2
110 s=RND*5+3
120 ORIGIN 320,-100
130 FOR x=-1000 TO 0 STEP s
140 MOVE 0,0
150 DRAW x,300:DRAW 0,600
160 MOVE 0,0
170 DRAW -x,300:DRAW 0,600
180 a$=INKEY$
```

```
190 IF a$<>" " THEN 30: REM Premere un tasto per interrompere il ciclo
200 NEXT x
210 GOTO 10
```

I due esempi precedenti illustrano semplici concetti matematici in modo colorato e piacevole. Entrambi praticamente eseguono somme di numeri casuali per far sì che ogni disegno sia diverso dal precedente; sullo schermo i risultati di tali operazioni appaiono come linee casuali.

Il CPC 464 risulta eccellente come 'foglio elettronico' sul quale tracciare diagrammi: uno degli esempi più classici è il disegno dell'andamento della funzione Seno. Il programma che segue svolge appunto questo compito:

```
 10 REM funzione seno
 20 MODE 2
 30 INK 1,21
 40 INK 0,0
 50 CLS
 60 DEG
 70 ORIGIN 0,200
 80 FOR n=0 TO 720
 90 y=SIN(n)
100 PLOT n*640/720,198*y,1
110 NEXT
```

Il comando PLOT della riga 100 è quello che in realtà traccia la funzione, accendendo un pixel sullo schermo per ogni volta che viene calcolata una coppia di coordinate nel ciclo FOR/NEXT (righe 80-110).

Il CPC 464 dispone di molti comandi semplici e potenti. Ad esempio, per aumentare l'effetto scenografico del programma basta aggiungere la riga seguente:

```
15 BORDER 6,9
```

Date nuovamente RUN. Il bordo ora si alterna tra i colori numero 6 e 9 con frequenza di lampeggio uguale a quella di default. Per fare in modo che il programma cicli continuamente su se stesso fino alla pressione del tasto [ESC] (una volta per sospendere e due volte per interrompere), aggiungete la riga:

```
120 GOTO 50
```

Osservate che il lampeggio del bordo non cessa con il termine dell'esecuzione del programma, perché le due cose sono totalmente indipendenti. Per interrompere il lampeggio del bordo e colorarlo di blu, cambiate la riga 15 in

```
15 BORDER 2
```

Ridate il RUN al programma e il lampeggio del bordo terminerà.

Per cambiare il colore con cui viene tracciata la funzione e quello del fondo, dovete modificare i comandi INK alle righe 30 e 40. Il listato dopo tutte le modifiche sarà:

```
 10 REM funzione seno
 15 BORDER 2
 20 MODE 2
 30 INK 1,21
 40 INK 0,0
 50 CLS
 60 DEG
 70 ORIGIN 0,200
 80 FOR n=0 TO 720
 90 y=SIN(n)
100 PLOT n*640/720,198*y,1
110 NEXT
120 GOTO 50
```

Con l'ultimo parametro del comando PLOT alla riga 100 si stabilisce di tracciare la curva con il colore associato all'INK di codice 1 nella riga 30. Controllate le specifiche del comando PLOT nell'elenco delle istruzioni del Capitolo 8 e comprenderete il significato di ognuno dei parametri di tale comando.

Guardando da vicino la curva tracciata, noterete che non è una linea continua, ma una spezzata composta da molti piccoli segmenti. Il più piccolo segmento rappresentabile sullo schermo è il pixel.

5.2.5 — Il cursore grafico e il disegno sullo schermo

Nel paragrafo precedente sono stati illustrati alcuni modo in cui è possibile tradurre dati numerici in forma grafica e vi è stata data la possibilità di provare molti comandi. Prima di passare a disegnare sullo schermo è però necessario fare alcune considerazioni in modo da non creare confusione.

Il primo punto da considerare è l'attuale stato della memoria. Il computer ricorda i colori selezionati per il bordo, il fondo e la penna, anche dopo l'esecuzione del comando NEW. Per riportare ogni cosa alle condizioni iniziali premete simultaneamente i tasti [CTRL] [SHIFT] [ESC]. Prima di eseguire questa operazione salvate sulla cassetta il programma eventualmente presente in memoria.

Digitate:

```
NEW: CLS
```

in questo modo avete cancellato il programma precedente. Ora eseguite il comando:

DRAW 100,100

Il comando DRAW traccia una linea dall'attuale posizione del cursore grafico sino al punto di coordinate specificate (100,100 in questo caso). Il cursore grafico non viene mai visualizzato: è solo una espressione concettuale utilizzata per indicare il punto da cui avrà inizio una successiva operazione grafica.

Per determinare la posizione del cursore grafico vengono utilizzate le funzioni XPOS e YPOS. Digitate:

PRINT XPOS

La risposta sarà

100

che in questo caso è la stessa ottenuta con la YPOS.

Quando il cursore raggiunge l'ultima riga dello schermo viene eseguito lo spostamento di una riga verso l'alto di tutto quanto rappresentato sullo schermo. Le coordinate del cursore grafico rimangono però immutate. Per provare, premete il tasto di spostamento del cursore verso il basso [↓] sino a quando lo schermo non risulta del tutto pulito e poi chiedete nuovamente sia la XPOS che la YPOS. La posizione del cursore grafico è immutata perché i suoi valori sono stati conservati in memoria.

Per tracciare una linea di un determinato colore è necessario inserire un nuovo parametro nel comando DRAW, come avviene per il PLOT.

Prima dovete specificare l'INK, ricordando che potete utilizzare solo il numero di INK e di colori disponibili nel Modo in cui vi trovate. Ecco un semplice esempio:

```
10 MODE 1
20 INK 0,10
30 ORIGIN 0,0
40 INK 1,26
50 INK 2,0
60 DRAW 320,400,1
70 DRAW 640,0,2
```

Quello che segue è un programma esplicativo che utilizza tutti i comandi sin qui menzionati e contiene alcuni nuovi concetti. Osservate come la prima riga

(10) selezioni i colori e i codici degli INK per ottenere l'effetto desiderato, indipendentemente da eventuali definizioni già presenti in memoria:

```
 10 INK 0,0:INK 1,26:INK 2,6:INK 3,18:BORDER 0
 20 PRINT "Questo Programma disegna Pattern"
 30 MODE 1:DEG
 40 PRINT"Pattern a 3,4 o 6 lati?";
 50 LINE INPUT p$
 60 IF p$="3" THEN sa=120:GOTO 100
 70 IF p$="4" THEN sa=135:GOTO 100
 80 IF p$="6" THEN sa=150:GOTO 100
 90 GOTO 50
100 PRINT:PRINT"Sto' calcolando..."
105 IF p$="3" THEN ORIGIN 0,-50,0,640,0,400 ELSE ORIGIN 0,0,0,640,0,400
110 DIM cx(5),cy(5),r(5),lc(5)
120 DIM np(5)
130 DIM px%(5,81),py%(5,81)
140 st=1
150 cx(1)=320:cy(1)=200:r(1)=80
160 FOR st=1 TO 4
170 r(st+1)=r(st)/2
180 NEXT st
190 FOR st=1 TO 5
200 lc(st)=0:np(st)=0
210 np(st)=np(st)+1
220 px%(st,np(st))=r(st)*SIN(lc(st))
230 py%(st,np(st))=r(st)*SIN(lc(st))
240 lc(st)=lc(st)+360/r(st)
245 IF (lc(st) MOD 60)=0 THEN PRINT".";
250 IF lc(st)<360 THEN 210
252 px%(st,np(st)+1)=px%(st,1)
254 py%(st, np(st)+1)=py%(st,1)
260 NEXT st
265 CLS:INK 1,2
270 st=1
280 GOSUB 340
290 LOCATE 1,1
300 EVERY 25,1 GOSUB 510
310 EVERY 15,2 GOSUB 550
320 EVERY 5,3 GOSUB 590
330 GOTO 330
340 REM
```

```
350 cx%=cx(st):cy%=cy(st):lc(st)=0
360 FOR x%=1 TO np(st)
370 MOVE cx%,cy%
380 DRAW cx%+px%(st,x%),cy%+py%(st,x%),1+(st MOD 3)
390 DRAW cx%+px%(st,x%+1),cy%+py%(st,x%+1),1+(st MOD 3)
400 NEXT x%
410 IF st=5 THEN RETURN
420 lc(st)=0
430 cx(st+1)=cx(st)+1.5*r(st)*SIN(sa+lc(st))
440 cy(st+1)=cy(st)+1.5*r(st)*COS(sa+lc(st))
450 st=st+1
460 GOSUB 340
470 st=st-1
480 lc(st)=lc(st)+2*sa
490 IF (lc(st) MOD 360)<>0 THEN 430
500 RETURN
510 ik(1)=1+RND*25
520 IF ik(1)=ik(2) OR ik(1)=ik(3) THEN 510
530 INK 1,ik(1)
540 RETURN
550 ik(2)=1+RND*25
560 IF ik(2)=ik(1) OR ik(2)=ik(3) THEN 550
570 INK 2,ik(2)
580 RETURN
590 ik(3)=1+RND*25
600 IF ik(3)=ik(1) OR ik(3)=ik(2) THEN 590
610 INK 3,ik(3)
620 RETURN
```

Appena iniziata l'esecuzione, il programma vi chiederà di inserire un numero (riga 40): rispondete 3 per ottenere il risultato il più velocemente possibile. Una volta inserita la risposta verrà visualizzato il messaggio 'Un attimo di pazienza' e per alcuni secondi sullo schermo comparirà unicamente una riga di punti per indicare che il programma si trova ancora nella fase di elaborazione dei dati.

I sottoprogrammi chiamati dalle righe 300-320 provocano il lampeggio dei colori con la frequenza determinata dal comando EVERY. Per rallentare la frequenza del lapeggio modificate le righe 300-320 come segue:

```
300 EVERY 250,1 GOSUB 510
310 EVERY 150,1 GOSUB 550
320 EVERY  50,3 GOSUB 590
```

Per rendervi conto di ciò che avete realmente fatto potete leggere nel Capitolo 8 la descrizione del comando EVERY, che è sicuramente una delle caratteristiche più interessanti del BASIC AMSTRAD. Quando l'esecuzione del programma viene sospesa tramite la pressione (UNA VOLTA SOLA) del tasto [ESC], il comando EVERY accoda le richieste di interrupt da soddisfare.

Per provare l'effetto, sospendete l'esecuzione del programma per pochi secondi e riprendetela premendo un tasto qualsiasi. Lo schermo lampeggerà in modo frenetico, in quanto tutti i comandi di temporizzazione accodati si trovano in attesa di esecuzione. Essendo limitato il numero di posti disponibili, la coda sarà presto saturata e pertanto non potrà accogliere nuovi comandi EVERY, se non nei posti lasciati liberi da comandi eseguiti.

5.3 — Le Finestre (WINDOW)

L'utente può aprire sino ad otto finestre di testo in cui scrivere caratteri e una finestra grafica nella quale disegnare. Cambiando Modo di visualizzazione si provoca la chiusura di tutte le finestre eventualmente aperte.
NB. Se la finestra di testo è in realtà equivalente a tutto lo schermo (come per default), le operazioni di scroll vengono eseguite via hardware, risultando perciò molto veloci. Se invece la finestra di testo è più piccola dell'intero schermo, le stesse operazioni devono essere eseguite dal software di base, risultando più lente.

Il comando WINDOW è utilizzato per definire i limiti di una finestra di testo e associarla ad uno degli 8 canali video disponibili. È possibile sovrapporre fra loro finestre, permettendo di riempire molto velocemente con un determinato colore intere zone dello schermo. Prima di iniziare ad esercitarvi nell'utilizzo delle finestre, digitate:

```
KEY 139, "mode 2: paper 0: ink 1,0: ink 0,9: list"+chr$(13)
```

In questo modo, qualunque sia la combinazione di colori ed il Modo in cui vi troviate, premendo il piccolo [ENTER] riporterete la situazione alla normalità senza perdere il programma presente in memoria. Il programma seguente disegna una serie di finestre sulla diagonale dello schermo:

```
10 MODE 0
20 FOR N=0 TO 7
30 WINDOW #N,N+1,N+6,N+1,N+6
40 PAPER #N,N+4
50 CLS #N
60 FOR C=1 TO 200: NEXT
70 NEXT
```

Questo programma illustra due concetti fondamentali: ogni finestra si sovrappone alla precedente e ogni messaggio viene visualizzato sul canale #0, a meno di un diverso indirizzamento. Prima di fare qualunque altra cosa digitate:

LIST

e il listato del programma verrà totalmente visualizzato nella finestra #0, spezzando le righe dove necessario. Ora provate:

LIST #5

e poi:

CLS #6

questo per mostrare come la finestra, e quindi il canale video, aperto per ultimo si sovrapponga a tutto il resto e come, nonostante il listato del programma sia stato inviato sul canale #5, il messaggio Ready del BASIC appaia nella finestra #0.

Inserendo il comando WINDOW SWAP alla riga 55:

55 IF N=3 THEN WINDOW SWAP 7,0

è possibile inviare il Ready del BASIC sul canale #7. Ridate il RUN e osservate. Elaborando ulteriormente questo semplice programma potrete comprendere ed apprezzare il modo in cui le Finestre operano ed interagiscono.

CAPITOLO 6
NOZIONI GENERALI SUL SUONO

Gli effetti sonori sono generati tramite un altoparlante interno al computer, per cui il controllo del volume dell'eventuale televisore domestico utilizzato insieme al Modulatore/Alimentatore MP1 deve essere posto al minimo.
Il livello del suono può essere regolato agendo sul controllo VOLUME posto sul lato destro del CPC 464. Il segnale sonoro può anche essere inviato alla presa ausiliaria in ingresso su un impianto stereo tramite l'uscita I/0 che si trova a sinistra sul retro del computer: in questo modo è possibile ascoltare in stereofonia il suono generato dal CPC 464.
Per poter sfruttare al massimo le notevoli capacità sonore del CPC 464, è opportuno soffermarsi sulle strutture di temporizzazione.

Argomenti trattati in questo capitolo:

- Periodi del tono
- Comando SOUND
- Inviluppo
- Code e sincronizzazione

Se la vostra unica aspirazione è quella di ottenere un semplice beep, allora limitatevi a battere PRINT CHR$(7) e restate in attesa del modesto risultato. Se invece intendete sfruttare una delle caratteristiche più interessanti del CPC 464, continuate nella lettura di questo Capitolo, che vi permetterà di impadronirvi degli elementi fondamentali del suono, insegnandovi a costruire una scala musicale e ad inventare vari tipi di strumenti.

6.1 — I principi del suono

Il suono generato da una nota standard monotona presenta alcune caratteristiche degne di nota:

1) tono e variazioni dello stesso nell'arco della durata della nota

2) volume e variazioni dello stesso nell'arco della durata della nota

3) lunghezza della nota

6.2 — Tono

In una nota musicale il tono è di gran lunga l'elemento più importante. Nella sua forma più semplice, la nota può essere rappresentata come una oscillazione sinusoidale descritta dai normali parametri di frequenza, periodo e ampiezza.

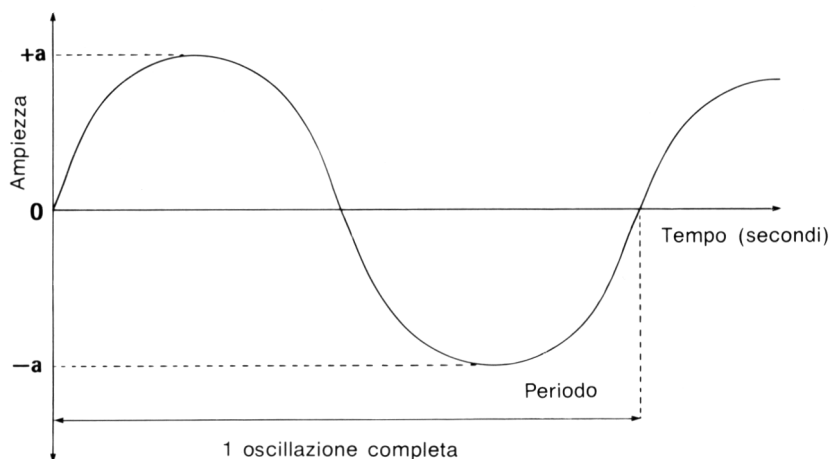



Figura 1 — Le caratteristiche di una nota musicale.

La frequenza rappresenta il numero di oscillazioni per secondo, mentre il periodo è misurato dall'intervallo di tempo intercorrente tra due oscillazioni successive. L'ampiezza è un attributo del volume non significativo nella definizione del tono della nota.

Fra frequenza e periodo esiste una semplice relazione espressa dalla formula:

frequenza (in Hz) = 1 / periodo (in sec).

La relazione tra la frequenza e il <periodo di tono> diventa nel comando SOUND:

frequenza (in Hz) = 125000 / <periodo di tono>.

Pertanto un <periodo di tono> di 1000 corrisponderà a una nota di frequenza pari a 125 Hz, mentre un <periodo di tono> di 125 produrrà una nota con frequenza di 1000 Hz (1kHz).

Sebbene il tono possa essere definito in uno qualunque dei due modi sopra riportati, il BASIC AMSTRAD sceglie il metodo del <periodo di tono>. Non lasciatevi ingannare dal fatto che il tono scende al salire del valore del periodo.

La gamma dei periodi di toni utilizzabili nel comando SOUND è molto estesa (da 0 a 4095) e deve essere definita tramite un numero intero, provocando alcuni errori di arrotondamento nella costruzione di una scala di note musicali. Ciò non comporta, peraltro, conseguenze avvertibili anche da parte degli uditi più sensibili (vedi Appendice VII).

Quando una nota viene generata su un vero strumento musicale il tono può variare, talora deliberatamente con il vibrato. Usando il comando ENT (ENvelope Tone = inviluppo del tono) è possibile stabilire un formato specifico per la struttura dei cambi di periodo durante ogni nota. Il comando SOUND, tramite uno dei parametri specificati, potrà successivamente richiamare lo <inviluppo di tono> definito con la ENT.

6.3 — Volume

Il volume è semplicemente la misura della forza del suono. La definizione del livello iniziale di volume per ciascun comando SOUND viene attuata tramite una relazione diretta:
incremento di valore = incremento del livello di volume, definito da un numero intero compreso tra 0 e 15.

Se avete già fatto dei tentativi con alcune note semplici, vi sarete accorti della relativamente povera fedeltà di riproduzione di una nota senza alcuna 'elaborazione'. Questo avviene perché con gli strumenti musicali convenzionali l'inizio (attacco) di ogni nota è individuato da un aumento di volume, così come la fine (caduta) è individuata da una diminuzione di volume.

Le diverse forme di attacco e caduta per ogni strumento musicale possono essere simulate sul computer tramite il comando ENV, che definisce lo <inviluppo di volume>.

6.4 — Lunghezza della nota

La lunghezza della nota è caratteristica fondamentale di ogni costruzione musicale. L'unità di misura della lunghezza della nota è la semiminima ed esistono minime, doppie crome ecc., che sono semplici multipli o sottomultipli dell'unità di base. Comunque, le note possono essere suonate a velocità variabile e la lunghezza base del tempo o periodo di riferimento per la semiminima standard deve risultare definito tramite il parametro <durata> del comando SOUND.

La <durata> è identificata da un numero intero, dove 1 unità corrisponde a 1/100 di secondo (cioè 100 = 1 sec). Da notare che la <durata> può essere anche determinata dalla lunghezza dello <inviluppo di volume>, in modo da evitare confusioni quando questo parametro risulta utilizzato.

6.5 — Altri suoni

Il computer può generare un 'rumore bianco'. Non si tratta di una nota musicale, ma di una interessante possibilità di aggiunta al sottofondo dei toni, che permette di ottenere variazioni o effetti speciali. Ad esempio, un'esplosione è essenzialmente rumore bianco con un controllo dello <inviluppo di volume>. Questo rumore ha come risultato di variare la frequenza casualmente nell'intorno di un periodo che viene definito nel comando SOUND come <periodo di rumore>. Caratteristica importante è che, pur esistendo tre canali separati con la possibilità di definire tre <periodi di tono>, si può definire un solo <periodo di rumore> che comparirà sulla combinazione di canali richiesta.

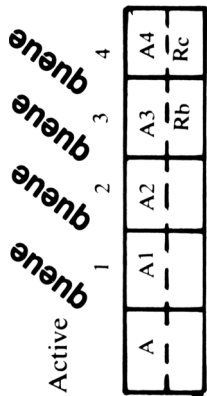
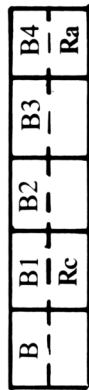
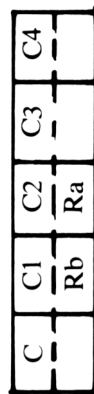
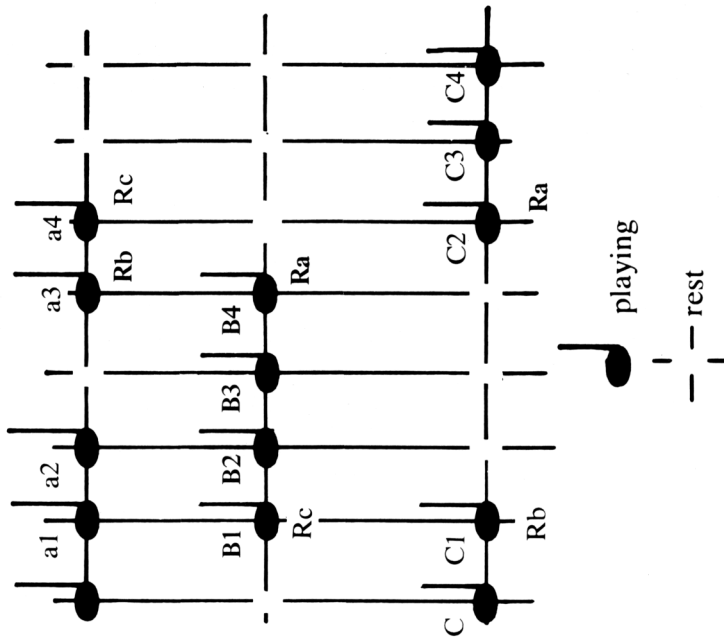
6.6 — Suoni multipli e canali

La maggior parte dei brani musicali è scritta con almeno due 'chiavi', di basso e di violino. Per rispettare questa norma il CPC 464 possiede tre canali sonori, rispettivamente chiamati A,B e C. I tre canali possono suonare indipendentemente l'uno dall'altro, ma è anche possibile inserire opportune temporizzazioni per ottenere, quando desiderato, la coincidenza dei suoni. Il parametro utilizzato nel comando SOUND per selezionare il canale è lo <stato canale>.

6.7 — Code

Ognuno dei tre canali A, B e C ha una coda di suoni in attesa di esecuzione. In detta coda possono trovare posto fino a cinque comandi SOUND diversi, di cui uno attivo e quattro in attesa. Il sistema operativo del CPC 464 è libero di eseguire altri compiti mentre vengono suonate le note in coda: esso ritornerà ad occuparsi del suono solo quando sarà necessario prelevare altri comandi SOUND.

6.8 — Stato del canale

Tramite il comando SQ è possibile conoscere lo stato in cui si trova un determinato canale, ottenendo informazioni sull'eventuale spazio libero nella coda, sulle situazioni degli 'appuntamenti' fra i canali e delle attese. Il comando ON SQ

Active
queue 1
queue 2
queue 3
queue 4
CHANNEL A
A
A1
A2
A3
Rb
A4
Rc
CHANNEL B
B
B1
Rc
B2
B3
B4
Ra
CHANNEL C
C
C1
Rb
C2
Ra
C3
C4
A–A4
B–B4
C–C4
Sound commands
Queue Head Status and Timing
a1
a2
a3
a4
Rb
Rc
B1
B2
B3
B4
Rc
Ra
C
C1
C2
C3
C4
Rb
Ra
playing
rest
Ra – rendezvous with channel A
Rb – rendezvous with channel B
Rc – rendezvous with channel C

GOSUB abilita un interrupt che permette di scoprire la presenza di buchi in una coda e riportare l'attenzione del computer alla sezione di generazione del suono del programma.

6.9 — Appuntamenti e attese

È possibile stabilire degli 'appuntamenti' fra due o più canali in modo da sincronizzare e rendere simultanea l'azione di più comandi SOUND. Questa caratteristica è molto utile per ripristinare la temporizzazione di note continue quando un tono contiene interruzioni o 'pause' non simultanee su entrambi i canali.

Lo stato di attesa (vedi 6.8) è importante dove è richiesta una sincronizzazione totale dei canali (ad esempio all'inizio di una nota), a volte con un tempo di ritardo, ed è necessario avere le code pronte. In questo caso lo stato di attesa viene rimosso tramite un comando RELEASE.

6.10 — Sequenza dei comandi per la generazione del suono

Forma del comando: SOUND G,H,I,J,K,L,M
Dove:

G : Stato del canale
H : Periodo di tono
I : Durata
J : Volume
K : Inviluppo di volume
L : Inviluppo di tono
M : Periodo di rumore

Tutti i parametri devono essere numeri interi e solo i primi due sono obbligatori. Gli altri parametri sono opzionali, ma hanno dei valori di default che verranno discussi di volta in volta.

6.10.1 — Descrizione dei parametri

G: Stato del canale
Valori compresi tra 0 e 255
Valore di default: nessuno (parametro obbligatorio)

Il CPC 464 ha la possibilità di eseguire tre suoni differenti nello stesso istante. Questo è permesso dai tre canali sonori (o code), chiamati A, B e C. Il parame-

tro viene dato sotto forma decimale, ma quando convertito in un intero su 8 bit, i bit posti a 1 indicano i seguenti comandi:

Decimale	Bit	Comando
1	0	invia il suono sul canale A
2	1	invia il suono sul canale B
4	2	invia il suono sul canale C
8	3	appuntamento con il canale A
16	4	appuntamento con il canale B
32	5	appuntamento con il canale C
64	6	stato di attesa
128	7	svuotamento della coda

alcuni esempi:

2 = invia questo suono sul canale B

5 = invia questo suono sui canali A e C

96 = 64 + 32 + 2 = invia questo suono sul canale B, fissando un appuntamento con il canale C e rimanendo poi in stato di attesa.

È importante ricordare che nel caso si voglia fissare un appuntamento fra due o più canali, è necessario segnalarlo al momento opportuno sullo <stato canale> di ogni canale.

Lo stato di attesa di una qualunque combinazione di canali ferma l'elaborazione di quel comando e congela la coda sotto di esso, fino a quando questa non viene liberata da un comando REALEASE (o 'svuotata' da un nuovo comando SOUND).

H: Periodo di tono
valori compresi tra 0 e 4095
valore di default: nessuno (parametro obbligatorio)

Definisce la frequenza con cui eseguire il suono. La frequenza può essere stabilita con la formula:

frequenza = 125000/periodo

Mettendo a 0 questo parametro non si specifica alcuna frequenza; questo è particolarmente utile quando si vuole generare solo rumore.

I: Durata
valori compresi tra –32768 e +32767
valore di default: 20

Per valori maggiori di 0 il parametro rappresenta la durata in centesimi di secondo. Quando uguale a 0, la durata è controllata dalla lunghezza dell'inviluppo di volume specificato.
Quando il valore fornito per la durata è minore di 0, questo indica il numero di volte che l'inviluppo di volume specificato deve essere ripetuto.

J: Volume
valori compresi tra 0 e 15 (tra 0 e 7 quando non è specificato alcun inviluppo di volume)
volume di default: 12 (4 dove non è specificato l'inviluppo di volume)

Indica il volume iniziale e può essere modificato dall'inviluppo di volume, se specificato.

K: Inviluppo di volume
valori compresi tra 0 e 15
valore di default: 0.

Tale parametro indica un inviluppo di volume predefinito tramite il comando ENV. L'inviluppo numero 0 non può essere modificato, in quanto si tratta di una definizione permanente, il cui livello di volume è costantemente posto a 2.

L: Inviluppo di tono
valori compresi fra 0 e 15.
valore di default: 0

Tale parametro indica un inviluppo di tono predefinito tramite il comando ENT. L'inviluppo numero 0 non può essere modificato in quanto si tratta di una definizione permanente il cui <periodo di tono> è costante.

M: Periodo di rumore
valori compresi tra 0 e 15
valori di default: 0

Specifica il rumore da aggiungere al suono. Utilizzando il valore di default non viene aggiunto alcun rumore. Ricordate che è possibile definire un solo periodo di rumore alla volta. Questo significa che tutti i canali specificati per il rumore riceveranno lo stesso rumore.

6.11 — Il comando ENV e gli inviluppi di volume

Per dare corpo e vita alla nota occorre utilizzare il comando ENV (ENvelope Volume = inviluppo di volume) tramite il quale si riesce a definire la forma d'onda

della nota da generare, attacco e caduta inclusi. È buona norma mettere su carta le vostre esigenze sotto forma di sequenze numeriche, prima di passare a descrivere al computer la forma d'onda. Il diagramma che segue chiarisce quello che intendiamo dire:

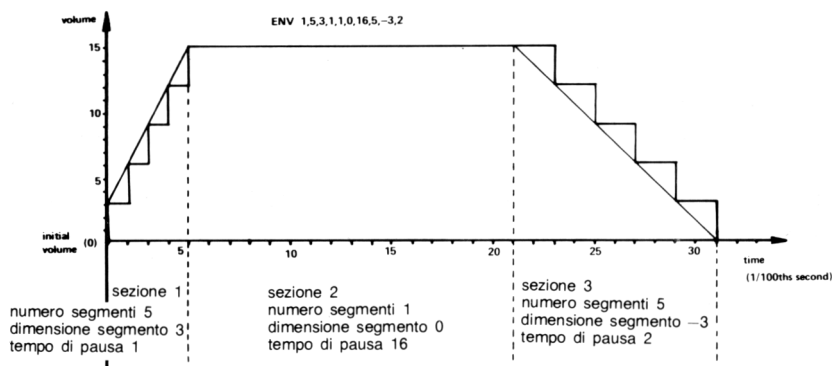



Figura 3 — Esempio di inviluppo di volume.

La forma d'onda deve essere tradotta in numeri che verranno successivamente forniti come parametri al comando ENV. A tal fine, si suddivida la curva in sezioni fino ad un massimo di cinque, ricordando che all'interno di ogni sezione la forma d'onda deve essere approssimata ad una linea retta. Ora si suddivida ogni sezione nel <numero segmenti> desiderato, ciascuno di lunghezza in tempo pari a <tempo di pausa> con 1 = 1/100 sec.

Tutti i segmenti appartenenti ad una sezione avranno lo stesso aumento/diminuzione in volume pari alla <dimensione segmento>. Possono esistere sezioni senza segmenti, nel qual caso si ottiene di riportare alla sezione successiva una semplice regolazione di volume.

N.B. Spesso è opportuno, quando si tenta di simulare uno strumento musicale, che il volume iniziale e finale di ogni nota sia pari a 0. In tal caso il volume iniziale definito nel comando SOUND sarà 0 e tutto il volume verrà successivamente aggiunto dall'inviluppo.

Forma del comando:

ENV N,P1,Q1,R1,P2,Q2,R2,P3,Q3,R3,P4,Q4,R4,P5,Q5,R5

N :	numero dell'inviluppo	valori	tra	1	e	15
P1/P5:	numero di segmenti (sezioni 1/5)	valori	tra	0	e	127
Q1/Q5:	dimensione segmento (sezioni 1/5)	valori	tra	−128	e	+127
R1/R5:	tempo di pausa (sezioni 1/5)	valori	tra	0	e	255

Il <numero inviluppo> è obbligatorio, così come obbligatoria è la presenza di almeno una sezione completa, mentre il numero delle sezioni è opzionale. Ad esempio, se si omettono le sezioni 4 e 5, automaticamente la forma d'onda risulterà completamente descritta da tre sole sezioni.

6.12 — Il comando ENT e gli inviluppi di tono

Il procedimento di costruzione è identico a quello già descritto per gli inviluppi di volume, mentre lo scopo è di creare piccole variazioni nella frequenza di una nota, quasi una forma di vibrato. Dopo aver deciso la forma della frequenza di una nota, disegnatela come spiegato al paragrafo precedente e dividetela in sezioni e segmenti, rifacendovi all'esempio che segue:

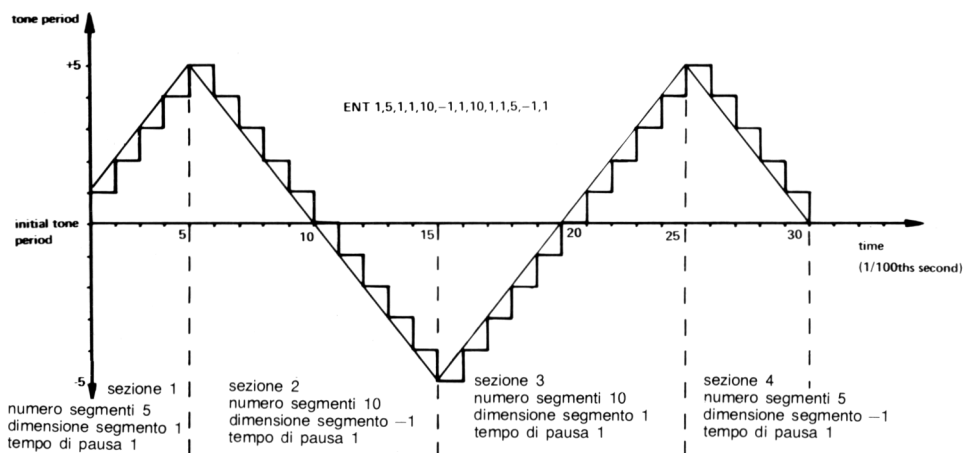



Figura 4 — Caratteristiche di un inviluppo di tono.

La differenza principale fra gli inviluppi di volume e gli inviluppi di tono consiste nel fatto che questi ultimi non hanno effetto sulla durata di una nota precedentemente definita vuoi con un comando SOUND vuoi con un inviluppo di volume. Se l'inviluppo di tono termina prima della nota, il periodo di tono costante definito nel comando SOUND coprirà il tempo rimanente. Se si oltrepassa la scala di tempo della nota le sezioni rimanenti non vengono considerate.

Per ripetere un inviluppo di tono per la durata del suono, utilizzate un numero di inviluppo negativo, ricordando però di indicarlo nel comando SOUND come positivo.

Forma del comando:

ENT S,T1,V1,W1,T2,V2,W2,T3,V3,W3,T4,V4,W4,T5,V5,W5

S	: numero dell'inviluppo	valori da 1 a 15 (– per ripetere)
T1/T5	: numero di segmenti (sezioni 1/5)	valori da 0 a + 127
VI/V5	: dimensione segmento (sezioni 1/5)	valori da –128 a + 127
W1/W5	: tempo di pausa (sezioni 1/5)	valori da 0 a 255 (1/100 sec)

S è obbligatorio, così come il completamento di ogni singola sezione.

Quando si definisce un <numero inviluppo>, tutte le eventuali precedenti regolazioni vengono annullate. L'assenza di sezioni nella descrizione di un inviluppo, sia di volume che di tono, ha come effetto di azzerare tutte le regolazioni relative a quel <numero inviluppo>.

6.13 — Funzioni e comandi associati

SQ (x), dove x rappresenta il numero di canale e può essere 1,2 o 4 (canale A,B o C).

Si tratta di una interrogazione fatta sul canale di numero specificato per conoscerne lo stato. Il risultato è un intero compreso tra 0 e 255. Il significato dei bit posti ad 1 si può desumere dalla tabella che segue:

Bit 0/2	numero di spazi liberi nella coda
Bit 3	appuntamento con il canale A
Bit 4	appuntamento con il canale B
Bit 5	appuntamento con il canale C
Bit 6	stato di attesa in testa alla coda
Bit 7	canale che sta suonando

I Bit da 3 a 6 sono già stati descritti nella sezione relativa allo stato del canale. L'unico altro compito di questo comando è quello di disabilitare l'interrupt provocato dalla ON SQ GOSUB, descritta di seguito.

ON SQ GOSUB
ON SQ(y) GOSUB <numero di riga>

dove y indica come al solito il numero di canale.

Questo comando abilita un interrupt per rilevare la presenza di un buco nella coda del canale indicato e rimanda alla subroutine che inizia alla riga specificata. Sia il comando SQ che il SOUND disabilitano l'interrupt. La subroutine termina normalmente con un RETURN. Tutti i canali hanno la stessa priorità.

Da notare che il verificarsi del primo interrupt disabilita automaticamente l'intercettazione del successivo, per cui sarà necessario riabilitare l'interrupt nella subroutine se si intende riutilizzare quest'ultima.

RELEASE
RELEASE z

dove z indica il numero del canale, o dei canali, e può assumere un valore compreso tra 1 e 7.

Come precedentemente descritto, è possibile mettere un canale in stato di attesa tramite un particolare comando SOUND. Il comando RELEASE serve per rimuovere questa situazione sui canali specificati e individuati tramite la decodifica della configurazione binaria di z.

CAPITOLO 7
STAMPANTI E JOYSTICK

Il CPC 464 può essere direttamente collegato a uno o due joystick e una stampante con interfaccia parallela Centronics.

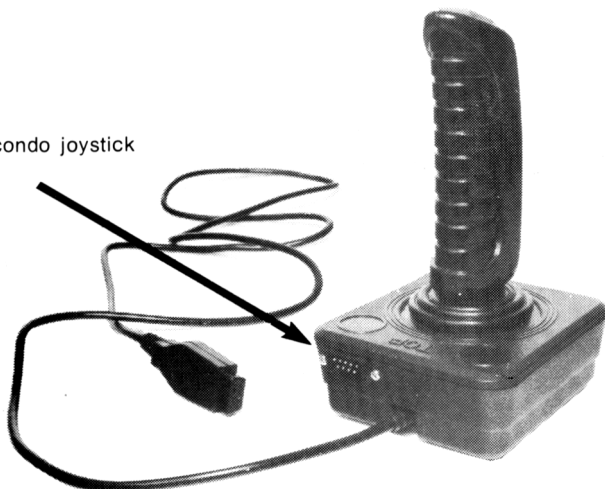
Argomenti trattati in questo capitolo:

- I joystick
- Le stampanti con interfaccia parallela
- Interfacciamento

Il joystick AMSOFT modello JY1 è un accessorio appositamente concepito per controllare agevolmente il movimento di oggetti sullo schermo e per 'sparare' al nemico quando il gioco lo richiede. La connessione avviene tramite la porta a 9 pin identificata dalla dicitura USER PORTS (I/O) sul retro del computer. Qualora il CPC 464 venga utilizzato con due joystick, il secondo dovrà essere collegato al primo per mezzo del connettore presente sulla base dello stesso. La piedinatura di tutte le porte di I/O del CPC 464 è riportata nell'Appendice V.

7.1 — I joystick

Il software di base del CPC 464 è in grado di gestire sino a due joystick, che vengono trattati come parte integrante della tastiera e il cui stato può essere rilevato tramite i comandi INKEY$ e INKEY, proprio come se si trattasse di tasti. Tenete presente che se il vostro joystick dispone di un solo bottone di fuoco (Fire), questo sarà identificato come 'Fire 2' nel corso del manuale.

È prevista anche una funzione apposita per rilevare direttamente lo stato dei joystick: con JOY(0) si legge lo stato del primo joystick, con JOY(1) quello del secondo. La funzione restituisce come risultato una configurazione binaria che rappresenta la posizione degli switch del joystick dopo l'ultima scansione della

tastiera. Dal momento che la scansione viene effettuata ogni cinquantesimo di secondo, il risultato ottenuto può considerarsi rappresentativo della posizione istantanea degli switch.

La tabella che segue riporta i valori restituiti dalla funzione JOY e i codici (colonna KEY) da utilizzare con la funzione INKEY per leggere lo stato del joystick. Da notare la colonna TASTO, che riporta i valori da utilizzare in una eventuale INKEY$ per l'interrogazione del secondo joystick, in alternativa alla JOY(1) o alla INKEY. Questo perchè il CPC 464 non può stabilire la differenza tra la posizione del joystick 2 e la pressione del TASTO corrispondente. In effetti, la tastiera può sostituire a tutti gli effetti il secondo joystick.

Primo Joystick	JOY(0)	KEY	Secondo Joystick	JOY(1)	KEY	TASTO
SU	1	72	SU	1	48	6
GIÙ	2	73	GIÙ	2	49	5
SINISTRA	4	74	SINISTRA	4	50	R
DESTRA	8	75	DESTRA	8	51	T
FIRE 2	16	76	FIRE 2	16	52	G
FIRE 1	32	77	FIRE 1	32	53	F

Utilizzando l'AMSOFT JY1, il secondo joystick è assolutamente identico al primo e si connette direttamente alla presa che si trova sulla base del primo.

Il connettore a 9 pin marcato USER PORTS (I/O) accetta anche joystick standard diversi dal JY1, anche se utilizzando questi ultimi non è possibile collegare

un secondo joystick, a meno che non venga interposto un adattatore particolare. In ogni caso, evitare di utilizzare un joystick standard come secondo assieme ad un JY1.

7.2 — Interfacciamento di stampanti

Al CPC 464 può essere connessa una stampante parallela conforme allo standard Centronics.

Il cavo stabilisce un collegamento uno a uno tra il connettore del CPC 464 e quello della stampante. I dettagli dell'interfaccia si trovano nell'Appendice V.

Il cavo deve essere assemblato in modo che il Pin 1 del CPC 464 venga connesso con il Pin 1 della stampante, il Pin 19 del computer con il medesimo della stampante ecc.; i Pin 18 e 36 della stampante non vengono collegati.

In particolare, il connettore del CPC 464 è strutturato in modo tale per cui il collegamento dei Pin avviene sempre e comunque tra quelli di numero corrispondente, indipendentemente dal tipo di cavo utilizzato. Per questo motivo, la numerazione dei Pin della fila inferiore del connettore lato computer parte da 19, invece che da 18, come sarebbe logico aspettarsi.

Il computer utilizza il segnale BUSY (pin 11) per sincronizzarsi con la stampante e rimane in stato di attesa se questa si trova OFF LINE.

Per utilizzare la stampante non sono necessari comandi di inizializzazione; l'ouput viene automaticamente inviato verso la stampante specificando nei comandi il canale 8. Ad esempio:

```
LIST #8
```

invia il listato del programma BASIC alla stampante, ammesso che il programma sia non protetto.

All'interno dei programmi è possibile inviare caratteri alla stampante utilizzando il comando PRINT nella forma:

```
PRINT #8,"Questo messaggio viene inviato alla stampante"
```

Molte stampanti provvedono automaticamente ad andare a capo quando l'ouput raggiunge la fine della linea di stampa: controllate sempre il manuale della stampante. Il BASIC AMSTRAD esegue automaticamente questa operazione secondo quanto definito con il comando WIDTH. La lunghezza della riga di stampa è per default di 132 caratteri, ma può essere ridefinita in relazione alle specifiche necessità.

Con il comando WIDTH 255 si indica al CPC 464 di non effettuare alcun controllo di fine linea e di affidare totalmente questo compito alla stampante. Il BA-

SIC conserva un contatore della posizione a cui si trova la testina di stampa; questo valore può essere rilevato tramite la funzione POS. Es.:

```
IF POS(#8) > 50 THEN GOTO 100
```

Il CPC 464 inserisce un carattere di avanzamento della carta (CHR$(10)) e uno di ritorno carrello (CHR$(13)) alla fine di ogni riga. La stampante normalmente contiene internamente alcuni switch atti a selezionare il formato dell'input: i valori di default risulteranno evidenti alla prima stampa.

7.3 — La stampa dei caratteri grafici

Il manuale tecnico della vostra stampante riporterà i codici di controllo ed il loro significato. Questi vengono solitamente inviati con il comando:

```
PRINT #8,CHR$(n)
```

Alcune stampanti possono avere un set di caratteri simile a quello del CPC 464 (vedi Appendice III), ma è difficile che a codici uguali corrispondano caratteri uguali, per cui sarà opportuno preparare una tabella di conversione tra il set di caratteri della stampante e quello dell'AMSTRAD.

L'interfaccia parallela del CPC 464, pur essendo stata concepita per il collegamento con stampanti a matrice di punti, è comunque utilizzabile con plotter, stampanti a margherita e a getto di inchiostro, grazie al rispetto dei protocolli di comunicazione standard.

CAPITOLO 8

IL BASIC AMSTRAD: GUIDA DI CONSULTAZIONE RAPIDA

Argomenti trattati in questo capitolo:

- Formato dei dati
- Caratteri speciali e loro significato
- Le istruzioni del BASIC AMSTRAD in ordine alfabetico

8.1 — Formato dei dati

Caratteri Speciali

& o &H	prefisso per le costanti esadecimali (base 16)
&X	prefisso per le costanti binarie (base 2)
:	separatore delle istruzioni sulla stessa riga
#	prefisso per l'identificatore di canale (stream)

Tipi di dati

Le stringhe possono essere lunghe da 0 a 255 caratteri e sono individuate come <stringa>. Il concatenamento di stringhe è ottenibile tramite l'uso dell'operatore '+', a patto che la stringa risultante sia sempre di lunghezza inferiore a 255 caratteri.

I dati numerici possono essere sia Interi che Reali. Gli interi sono compresi in un campo che va da —32768 a 32767; i reali hanno una precisione di nove cifre nel campo ±1.7E+38 con il valore minimo positivo pari a circa 2.9E—39.

Gli indicatori del tipo di dato sono:

% Intero
! Reale
$ Stringa

Si definisce <espressione numerica> una qualunque espressione che fornisca un risultato numerico: può trattarsi semplicemente di numeri, o di variabile numerica, oppure del risultato di un'operazione su variabili numeriche. Praticamente, tutto ciò che non è <stringa>.

Si definisce <canale> una <espressione numerica> che identifica lo schermo, la stampante o la cassetta verso cui il testo deve essere 'incanalato'.

Per <argomento improprio> intendiamo significare che una espressione numerica ha fornito un valore esterno al campo di variabilità consentito o che il parametro di una istruzione è in qualche modo non valido per detta istruzione e pertanto non accettato dal BASIC.

ISTRUZIONI

Le istruzioni del BASIC AMSTRAD sono elencate utilizzando il formalismo:

ISTRUZIONE
Sintassi/Funzione
Esempio
Descrizione
ISTRUZIONI associate

IMPORTANTE:

Per Istruzioni intendiamo

COMANDI: operazioni che vengono eseguite direttamente.
oppure
FUNZIONI: operazioni che costituiscono gli argomenti di una espressione.

Parentesi

Le parentesi rotonde ( ) sono parte integrante di un comando o di una funzione; gli altri tipi di parentesi utilizzate nella descrizione delle istruzioni sono inseriti solo a titolo esplicativo e, pertanto, non fanno parte della sintassi delle istruzioni. Il significato è:

[ ] delimitano i parametri opzionali
< > delimitano i tipi di espressione descritti nel seguito.

Apici

Solo i doppi apici " " fanno parte del BASIC AMSTRAD. I singoli apici ' ' vengono utilizzati per sottolineare alcuni aspetti della descrizione e pertanto non devono essere considerati nella sintassi delle istruzioni.

Inserimento delle Istruzioni

Il BASIC converte in lettere maiuscole tutte le Istruzioni inserite, anche quelle battute in minuscolo. Gli esempi mostrano tutte le istruzioni in maiuscolo in quanto tale è la forma con cui appaiono nel listato del programma. Inserendo le Istruzioni in minuscolo si è facilitati nella ricerca degli errori di battitura, in quanto le Istruzioni scritte scorrettamente rimmarranno in minuscolo sullo schermo.

Le Istruzioni devono essere delimitate da spazi in quanto il BASIC AMSTRAD permette di utilizzare le Istruzioni nei nomi delle Variabili. Ad esempio: end2 e LISTCODE sono nomi di variabili accettati dal BASIC AMSTRAD.

ABS

ABS (<espressione numerica>)

PRINT ABS(–67.98)
 67.98

FUNZIONE: Restituisce il valore assoluto dell'espressione fornita come parametro. Essenzialmente serve per trasformare i numeri negativi in positivi.

Istruzioni associate: SGN

AFTER

AFTER <espr. intera> [,<espr. intera>] GOSUB <numero di riga>

AFTER 200,2 GOSUB 320

COMANDO: Chiama una subroutine una volta trascorso un certo lasso di tempo. La prima <espressione> indica la durata del ritardo in cinquantesimi di secondo; la seconda <espressione> (valore tra 0 e 3) specifica quale dei quattro timer disponibili deve essere utilizzato.

Istruzioni associate: EVERY, REMAIN

ASC

ASC (<stringa>)

PRINT ASC ("X")
 88

FUNZIONE: Restituisce il codice ASCII del primo carattere della <stringa>.
Istruzioni associate: CHR$

ATN

ATN (<espressione numerica>)

PRINT ATN (1)
0.785398163

FUNZIONE: Calcola l'arcotangente della <espressione numerica> specificata. Il risultato è espresso normalmente in radianti, tra −PI/2 e PI/2.

Istruzioni associate: SIN, COS, TAN, DEG, RAD

AUTO

AUTO [<numero di riga>][,<incremento>]

AUTO 100,50

COMANDO: Genera automaticamente i numeri di riga. Il <numero di riga> indica il primo numero di riga da generare, nel caso in cui si vogliano aggiungere righe ad un programma già esistente. Sia il valore dell'<incremento> fra i numeri di riga, che il primo numero di riga da generare sono pari a 10, se non specificati.

Quando il numero generato è quello di una riga già esistente, che verrebbe perciò sovrascritta, il BASIC inserisce un asterisco * come avvertimento dopo il numero di riga generato.

BIN$

BIN$ (<espressione intera senza segno> [,<espressione intera>])

PRINT BIN$(64,8)
 01000000

FUNZIONE: Produce una stringa di cifre binarie rappresentante il valore della <espressione intera senza segno>, aggiungendo zeri a sinistra sino a raggiungere almeno il numero di cifre specificato dalla <espressione intera>

Istruzioni associate: HEX$, STR$

BORDER

BORDER <colore> [,<colore>]

BORDER 3,2

COMANDO: Modifica il colore del bordo sullo schermo. Se vengono specificati due colori, il bordo si alternerà fra i due con la frequenza specificata dal comando SPEED INK. I codici dei colori del bordo vanno da 0 a 26.

Istruzioni associate: SPEED INK

CALL

CALL <indirizzo>[,<lista di parametri>]
CALL &BD19

COMANDO: Permette di chiamare da BASIC una routine in linguaggio macchina. È da utilizzare con cautela, soprattutto da parte dei non esperti. La CALL dell'esempio è abbastanza 'innocua', in quanto attende il ritorno del pennello video, permettendo, nel frattempo, di mettere in ordine i caratteri in movimento sullo schermo per creare effetti di animazione.

CAT

CAT

CAT

COMANDO: Provoca l'inizio della lettura della cassetta e la visualizzazione dei nomi di tutti i file trovati. Non produce alcun effetto sul programma presente in memoria, perciò può essere utilizzata per verificare l'effettivo salvataggio del programma tuttora presente in memoria, prima che questo venga modificato o cancellato. Il Comando chiede di far partire il registratore e, quando viene trovato un programma, risponde con il messaggio:

NOMEPROGRAMMA Numero di blocco Tipo OK

Il Tipo indica il modo di registrazione:

$ Programma BASIC
% Programma BASIC protetto
* File testo di tipo ASCII
& File binario

Nel caso che il programma non sia stato prodotto dal BASIC possono apparire caratteri diversi da quelli sopraelencati.

Istruzioni associate: LOAD, RUN, SAVE

CHAIN - CHAIN MERGE

CHAIN <nome file>[,<numero di riga>]
CHAIN MERGE <nome file>[—<numero di riga>]
[,DELETE <numero di riga—numero di riga>]

CHAIN "TEST", 350

COMANDO: CHAIN carica un programma dalla cassetta in memoria, cancellando il programma esistente. CHAIN MERGE concatena un programma proveniente da cassetta con quello attualmente in memoria. Il <numero di riga> indica da quale linea deve iniziare l'esecuzione del programma dopo il concatenamento. In mancanza del <numero di riga> l'esecuzione inizia dalla riga di programma con il numero più basso.

Se non viene specificato alcun <nome file>, il BASIC cercherà di leggere e concatenare il primo file valido incontrato sul nastro. Se il primo carattere del <nome file> è !, allora tale carattere verrà cancellato dal <nome file> e tutti i normali messaggi di lettura da cassetta verranno soppressi.

CHAIN MERGE conserva il valore di tutte le variabili, tranne le funzioni definite dall'utente e i file aperti che vengono eliminati. L'opzione ON ERROR GOTO è ignorata, viene eseguito un RESTORE e le condizioni DEFINT, DEFREAL e DEFSTR sono resettate; tutti i cicli FOR e WHILE attivi sono ignorati, così come i comandi GOSUB. I file protetti non vengono concatenati.

Istruzioni associate: LOAD, MERGE

CHR$

CHR$ (<espressione numerica>)

```
PRINT CHR$(100)
 d
```

FUNZIONE: Ritorna il carattere di codice specificato appartenente al set di caratteri del CPC 464 (Vedi Appendice III).

Istruzioni associate: ASC, LEFT$, RIGHT$, MID$, STR$

CINT

CINT (<espressione numerica>)

```
10 n=578.76543
20 PRINT CINT(n)

RUN
 579
```

FUNZIONE: Arrotonda il valore fornito come <espressione numerica> nel campo dei numeri interi.

Istruzioni associate: CREAL, INT, FIX, ROUND, UNT

CLEAR

CLEAR

CLEAR

COMANDO: Azzera tutte le variabili e i file.

CLG

CLG [<numero ink>]

CLG

COMANDO: Pulisce la pagina grafica, riempiendola con il colore del <numero ink>.

Istruzioni associate: CLS, ORIGIN

CLOSEIN

CLOSEIN

CLOSEIN

COMANDO: Chiude il file di input da cassetta. Comandi come NEW e CHAIN MERGE ignorano ogni file aperto.

Istruzioni associate: OPENIN, CLOSEOUT

CLOSEOUT

CLOSEOUT

CLOSEOUT

COMANDO: Chiude il file di output su cassetta.

Istruzioni associate: OPENOUT, CLOSEIN

CLS

```
CLS [#<canale>]

CLS
```

COMANDO: Pulisce la finestra di schermo specificata, utilizzando il colore di fondo della stessa.

CONT

```
CONT

CONT
```

COMANDO: Riprende l'esecuzione del programma dopo che questa è stata interrotta con *Break*, STOP o END, a patto che il programma non sia stato modificato.

COS

```
COS(<espressione numerica>)

?COS (34)
-0.848570274
```

e

```
deg:?COS(34)
 0.829037573
```

FUNZIONE: Calcola il coseno della <espressione> data. L'<espressione numerica> fornisce l'angolo in radianti, a meno che non sia stato eseguito il comando DEG.

Istruzioni associate: SIN, TAN, ATN, DEG, RAD

CREAL (<espressione numerica>)

```
5 DEFINT n
10 n=75.765
20 d=n/34.6
30 ? d
40 ? CREAL (n)
50 ? n/55.4

run
```

```
2.19653179
76
1.37184116
Ready
```

FUNZIONE: Converte il valore dell'<espressione numerica>, normalmente intera, in notazione reale.

Istruzioni associate: CINT

DATA

```
DATA <lista di costanti>
```

```
10 DIM NOMI$(3)
20 DIM COGN$(3)
30 FOR I=1 TO 3
40 READ NOMI$(I)
50 READ COGN$(I)
60 PRINT "NOME ";NOMI$(I); " COGNOME ";COGN$(I)
70 NEXT
80 DATA Paolo,Bianchi,Marco,Rossi,Anna,Gentile
```

COMANDO: Specifica le costanti che saranno assegnate alle variabili dichiarate con l'istruzione READ. Le costanti devono essere dello stesso tipo delle variabili utilizzate per la lettura. Un'istruzione DATA può essere posta in qualunque punto del programma.

Istruzioni associate: READ, RESTORE

DEF FN

```
DEF FN <nome>[(<parametri formali>)] = <espressione>
```

```
10 DEF FNinteressi (capitale) = 1.14 * capitale
20 INPUT "Qual'è il tuo capitale "; capitale
30 PRINT "L'ammontare totale dopo un anno è di "; FNinteressi(capitale)
```

COMANDO: Permette di definire funzioni semplici, che possono essere richiamate all'interno del programma allo stesso modo delle funzioni intrinseche del BASIC, come la COS. La DEF FN non deve essere inserita nel corpo del programma.

DEFINT - DEFSTR - DEFREAL

DEFtipo <campo>

```
DEFINT I-N
DEFSTR A,W-Z
DEFREAL C,D
```

COMANDO: Definiscono, rispettivamente come Intere, Alfanumeriche e Reali, tutte le variabili il cui nome inizia con una delle lettere comprese nel <campo> specificato. Le lettere iniziali possono essere maiuscole o minuscole.

Istruzioni associate: LOAD, RUN, CHAIN, NEW, CLEAR

DEG

DEG

```
DEG
```

COMANDO: Fissa la misurazione degli angoli, normalmente espressa in radianti, in gradi. Questo, sino all'esecuzione di una CLEAR, di una RAD o al caricamento di un nuovo programma.

Istruzioni associate: RAD

DELETE

DELETE <numero di riga> – <numero di riga>

```
DELETE 100 - 200
```

COMANDO: Elimina dal programma tutte le righe il cui numero è compreso tra gli estremi specificati. Usare con cautela.

Istruzioni associate: NEW

DI

DI

```
10 CLS
20 TAG
30 EVERY 10 GOSUB 100
40 X1=RND*320:X2=RND*320
50 Y=200+RND*200
60 FOR X=320-X1 TO 320+X2 STEP 2
```

```
70 DI:PLOT 320,0,1:MOVE X-2,Y:PRINT " ";:MOVE X,Y:PRINT "#";:EI
80 NEXT
90 GOTO 40
100 MOVE 320,0
110 DRAW X+8,Y-16,0
120 RETURN
```

COMANDO: Disabilita gli Interrupt, ad esclusione del *Break*, sino a quando non vengono riabilitati esplicitamente con una EI, oppure implicitamente dalla RETURN di chiusura di una subroutine chiamata su Interrupt.

È utilizzato quando non si vogliono interruzioni nel programma, in situazioni in cui, ad esempio, due differenti routine all'interno del medesimo programma si contendono l'utilizzo di una risorsa. Nell'esempio, il programma principale e il sottoprogramma chiamato dall'interrupt della riga 30 competono per l'utilizzo della pagina grafica.

Istruzioni associate: EI

DIM

DIM < lista di variabili con indice>

```
10 CLS:PRINT "Inserisci 5 nomi ....":PRINT
20 DIM B$(5)
30 FOR N=1 TO 5
40 PRINT "NOME "N;
50 INPUT B$(N)
60 NEXT
70 FOR N=1 TO 5
80 PRINT B$(N);" hai già comprato l'AMSTRAD CPC 464 ?"
90 NEXT
```

COMANDO: Alloca lo spazio necessario per contenere tutti gli elementi del vettore (5 nell'esempio). Nel caso l'indice massimo venga omesso, il BASIC lo pone automaticamente pari a 10. Non è possibile variare, pena l'errore, la dimensione di una variabile con indice, una volta dichiarata esplicitamente o assegnata implicitamente.

Una <variabile con indice> è tale per cui uno stesso nome di variabile può assumere tanti valori quanti sono gli elementi che ne definiscono la dimensione. Come esemplificazione pratica, si può pensare ad un parcheggio individuato da un nome (B$ nell'esempio) con ogni singolo posto identificato con un numero (1,2,3,4 e 5 nell'esempio).

Istruzioni associate: ERASE

DRAW

DRAW <coordinata x>,<coordinata y>[,<colore>]

DRAW 200,200,13

COMANDO: Traccia un segmento dalla posizione corrente del cursore grafico sino al punto di coordinate assolute specificate (200,200 nell'esempio). Le coordinate rimangono invariate nei 3 differenti Modi grafici. Altri esempi si trovano nel Capitolo 5.

Istruzioni associate: DRAWR, PLOT, PLOTR, MOVE, MOVER, TEST, TESTR, XPOS, YPOS, ORIGIN

DRAWR

DRAWR <spostamento sulle x>,<spostamento sulle y> [,<colore>]

DRAWR 200,200,13

COMANDO: Traccia un segmento dalla posizione corrente del cursore grafico sino al punto le cui coordinate sono calcolate prendendo come 0,0 la posizione attuale del cursore stesso.

Istruzioni associate: DRAW, PLOT, PLOTR, MOVE, MOVER, TEST, TESTR, XPOS, YPOS, ORIGIN

EDIT

EDIT <numero di riga>

EDIT 110

COMANDO: Permette di modificare una riga di programma già esistente. Vedere il Capitolo 1.4.

Istruzioni associate: LIST

EI

EI

EI

COMANDO: Riabilita gli interrupt, se disabilitati dall'istruzione DI.

Istruzioni associate: DI

END

END

END

COMANDO: Indica la fine del programma e non è obbligatorio. END chiude tutti i file da e verso cassetta e ritorna al modo diretto. I suoni in coda di attesa non vengono persi.

Istruzioni associate: STOP

ENT

ENT <numero inviluppo>[,<sezione inviluppo>]*

```
10 ENT 1,100,2,20
20 SOUND 1,100,1000,4,0,1
```

COMANDO: Durante la generazione del segnale sonoro è possibile variare il tono del suono secondo quanto definito dall'inviluppo di tono. Vedere il Capitolo 6 e l'Appendice VII per una completa descrizione del suono e delle operazioni ad esso connesse.

Il <numero inviluppo> è una <espressione intera>, con valore assoluto compreso fra 0 e 15, che identifica il numero dell'inviluppo da definire. Se il <numero inviluppo> è negativo l'inviluppo viene ripetuto.

Si possono specificare fino a 5 <sezioni inviluppo>, ciascuna delle quali può assumere una delle forme:

<numero segmenti>,<dimensione segmento>,<tempo di pausa>
oppure:= <periodo di tono>,<tempo di pausa>.

La prima forma definisce una variazione incrementale relativa al <periodo di tono> corrente. La seconda forma specifica un valore assoluto per il <periodo di tono>. Il significato dei vari parametri è il seguente:

<numero segmenti>, compreso fra 0 e 239, indica il numero di segmenti che compongono la sezione.

<dimensione segmento> rappresenta la quantità di cui variare il <periodo di tono> per ogni segmento della sezione. Deve essere un intero compreso fra −128 e +127.

<tempo di pausa> indica l'intervallo di tempo, espresso in centesimi di secondo, che deve trascorrere fra un segmento e l'altro. Il valore deve essere compreso fra 0 e 255, dove 0 è considerato come 256.

<periodo di tono> fornisce il nuovo valore per il periodo. Deve essere un intero compreso fra 0 e 4095.

Il comando SOUND definisce il <periodo di tono> iniziale e può riferirsi ad uno qualsiasi dei 15 inviluppi di tono. Se non c'è inviluppo, o viene utilizzato un inviluppo non precedentemente definito, il tono rimane costante per tutta la durata del suono.
L'inviluppo di tono non ha effetto sulla durata del suono. Se il suono termina prima della fine dell'inviluppo i segmenti rimanenti vengono semplicemente ignorati.
Un inviluppo di tono con <numero inviluppo> negativo viene ripetuto per tutta la durata del suono.
Ogni definizione di inviluppo di tono annulla la precedente. Modificare un inviluppo mentre il suono che lo utilizza è attivo può causare effetti indeterminati, ma interessanti.
Specificando un inviluppo privo di sezioni si cancellano tutti i precedenti valori. Ogni successivo richiamo di questo inviluppo utilizzerà i valori di default.

Istruzioni associate: ENV, SOUND

ENV

ENV <numero inviluppo>[,<sezione inviluppo>]*

```
10 ENV 1,100,2,20
20 SOUND 1,100,1000,4,1
```

COMANDO: Durante la generazione del segnale sonoro è possibile variare il volume del suono tramite questo comando. Un inviluppo di volume consiste di:

<numero segmenti>,<dimensione segmento>,<tempo di pausa> che definiscono le variazioni di volume, specificando un <numero segmenti> (tra 0 e 127), la <dimensione segmento> (tra −128 e +127) e il <tempo di pausa> fra un passo e l'altro in centesimi di secondo (tra 0 e 255).
Il <numero inviluppo> è un intero compreso fra 0 e 15, che individua il numero dell'inviluppo di volume da definire.

È possibile utilizzare sino a 5 <sezioni di inviluppo>. Ognuna può assumere la forma:

<numero segmenti>, <dimensione segmento>, <tempo di pausa>

oppure

<inviluppo via hardware>,<periodo di inviluppo>

La prima forma specifica una sezione di inviluppo controllata da software, dove i parametri sono:

<numero segmenti> indica il numero di segmenti che compongono la sezione. Valori compresi fra 0 e 127.

<dimensione segmento> indica l'entità della variazione nell'ampiezza di ciascun segmento. Valori compresi fra −128 e +127.

oppure: se il <numero segmenti> è posto a 0, rappresenta il valore assoluto dell'ampiezza.

<tempo di pausa> indica l'intervallo di tempo, espresso in centesimi di secondo, tra un segmento e l'altro. Valori compresi fra 0 e 255, dove 0 è trattato come 256.

La seconda forma specifica una sezione di inviluppo direttamente sotto il controllo dell'hardware, dove:

<inviluppo via hardware> è il valore da porre nel registro della forma d'onda dell'inviluppo (registro 15, ottale).

<periodo di inviluppo> è il valore da porre nei registri del periodo di inviluppo (registri 13 e 14, ottale).

L'inviluppo via hardware non ha associato alcun tempo di pausa e pertanto le sezioni dell'inviluppo vengono eseguite immediatamente, una di seguito all'altra. È comunque consigliabile che il segmento successivo abbia una pausa di lunghezza appropriata. Nel caso in cui non esista un segmento successivo, viene assunta una pausa di due secondi.
Il comando SOUND regola il volume iniziale e può riferirsi ad uno qualsiasi dei 15 inviluppi di volume disponibili. Se non c'è inviluppo, o ci si riferisce ad un inviluppo non definito, il volume rimane costante per tutta la durata del suono.
Specificando una <dimensione segmento> uguale a 0, con un <numero segmenti> diverso da 0, si ottiene il mantenimento del livello di volume corrente.
Per il resto vale quanto detto circa l'inviluppo di tono (vedi paragrafo precedente).

Istruzioni associate: ENT, SOUND

EOF

```
EOF

PRINT EOF
-1
```

FUNZIONE: Controlla lo stato del file in input da cassetta. Restituisce −1 (vero) se è stata raggiunta la fine del file, altrimenti 0 (falso).

Istruzioni associate: OPENIN

ERASE

ERASE <elenco nomi di variabili>

ERASE A,B$

COMANDO: Rilascia la memoria occupata dalle variabili specificate nell'elenco e la rende disponibile per altri usi.

ERR - ERL

ERR
ERL

```
10 CLS
20 ON ERROR GOTO 1000
30 DATA MARIA,ANNA,EMMA,PIERO,PAOLO
40 READ A$
50 PRINT A$
60 GOTO 40
70 REM inizio routine di trattamento dell'errore
1000 IF ERR = 4 THEN PRINT "Non ho più DATA da leggere"
1010 IF ERL<70 AND ERL>20 THEN PRINT "nelle righe 30-60"
1020 END
```

VARIABILI: Queste variabili di sistema vengono utilizzate nei sottoprogrammi di gestione dell'errore, per scoprire il tipo di errore e il numero della riga a cui si è verificato. Vedere l'elenco dei messagi d'errore nell'Appendice VIII.

Istruzioni associate: ON ERROR, ERROR

ERROR

ERROR <espressione intera>

ERROR 17

COMANDO: Provoca l'errore il cui codice è specificato come <espressione intera>. Nel caso che l'errore sia uno di quelli riconosciuti dal BASIC, il computer si comporta di conseguenza; codici d'errore diversi possono essere utilizzati per scopi particolari.

Istruzioni associate: ON ERROR, ERR, ERL

EVERY

EVERY <esp. intera>[,<esp. intera>] GOSUB <numero di linea>

EVERY 500,2 GOSUB 50

COMANDO: Il CPC 464 mantiene internamente un orologio in tempo reale. Il comando EVERY permette ad un programma BASIC di richiamare sottoprogrammi ad intervalli regolari. Sono disponibili quattro timer, identificati dalla seconda <espressione intera> (valore compreso fra 0 e 3). Ad ogni timer può essere associato un sottoprogramma.

Istruzioni associate: AFTER, REMAIN

EXP

EXP (<espressione numerica>)

PRINT EXP (6.876)
 968.743625

FUNZIONE: Eleva alla potenza dichiarata nella <espressione numerica> la costante 'E', che vale approssimativamente 2.7182818 e che costituisce la base dei logaritmi naturali.

Istruzioni associate: LOG

FIX

FIX (<espressione numerica>)

PRINT FIX (9.99999)
 9

FUNZIONE: La FIX tronca la parte della <espressione numerica> che si trova alla destra della virgola decimale, fornendo un risultato intero.

Istruzioni associate: CINT, INT, ROUND

FOR

FOR <variabile>=<val.iniziale> TO <val.finale> STEP <incremento>

FOR DAY = 1 TO 5 STEP 2

COMANDO: Indica l'inizio di un ciclo FOR/NEXT. La parte di programma compresa tra le due istruzioni viene eseguita sino a quando la <variabile> di control-

lo non assume il <valore finale>. Se non specificato l'<incremento> viene automaticamente considerato 1.

Istruzioni associate: NEXT, WHILE

FRE

FRE (<espressione numerica>)

FRE (<stringa>)

PRINT FRE (0)

PRINT FRE (" ")

FUNZIONE: Stabilisce la quantità di memoria ancora disponibile per l'utente. La forma FRE (" ") tiene conto, nello stabilire la quantità di memoria utilizzabile, anche di eventuali spazi sprecati.

GOSUB

GOSUB <numero di riga>

GOSUB 210

COMANDO: Salta al sottoprogramma che parte dal <numero di riga> specificato. Vedere anche RETURN.

Istruzioni associate: RETURN

GOTO

GOTO <numero di riga>

GOTO 90

COMANDO: Continua l'esecuzione partendo dal <numero di riga> specificato.

HEX$

•HEX$ (<espressione intera senza segno>)[,<espressione intera>]

PRINT HEX$(65534)
FFFE

FUNZIONE: Converte l'<espressione intera senza segno> in notazione esadecimale (vedi Appendice II).

Istruzioni associate: BIN$, STR$

HIMEM

HIMEM

?HIMEM
43903

VARIABILE: Restituisce l'indirizzo della più alta locazione di memoria accessibile al BASIC e può essere normalmente utilizzata in qualsiasi espressione numerica.

Prima di modificare il valore dell'indirizzo massimo di memoria accessibile, tramite il comando MEMORY, è consigliabile eseguire l'assegnazione mm = HIMEM. In questo modo sarete poi in grado di tornare al precedente valore tramite il comando MEMORY mm.

Istruzioni associate: FRE, MEMORY

IF

IF

IF <espressione logica> THEN <comando> [ELSE <comando>]

IF <espressione logica> GOTO <numero di linea> [ELSE <comando>]

IF A>B THEN A=C ELSE A=D

IF A>B GOTO 1000 ELSE 300

COMANDO: È estremamente utile per controllare l'esecuzione del programma. La <espressione logica> viene risolta e, nel caso risulti vera, si esegue il ramo THEN o GOTO. Se la <espressione logica> risulta falsa, il programma passa ad eseguire il <comando> successivo alla ELSE, oppure procede in sequenza nel caso che quest'ultima manchi. Non è possibile inserire nella stessa riga in cui si trova una IF altre istruzioni che non dipendano da questa.
Istruzioni associate: THEN, ELSE, GOTO, OR, AND, WHILE

INK

INK <inchiostro>,<colore>[,<colore>]

INK 0,1

COMANDO: Per <inchiostro> si intende il codice del colore con il quale vengono rappresentati i caratteri sullo schermo. A seconda dell'attuale Modo di visualizzazione, l'utente dispone di un certo numero di <inchiostri>. Il colore, o

colori, utilizzati per un <inchiostro> possono essere modificati da un comando INK, secondo i codici riportati nella tabella dei colori dell'Appendice VI.

Istruzioni associate: PEN, PAPER

INKEY

INKEY (<espressione intera>)

```
10 CLS:IF INKEY(55)=32 THEN 30 ELSE 20
20 CLS:GOTO 10
30 PRINT "Stai premendo [SHIFT] e V"
40 FOR T=1 TO 1000:NEXT:GOTO 10
```

FUNZIONE: Viene interrogata la tastiera per determinare l'eventuale tasto premuto. La scansione viene ripetuta ad intervalli di un cinquantesimo di secondo. Questa funzione è particolarmente utile per riconoscere una scelta del tipo SI/NO. I tasti [SHIFT] e [CTRL] sono identificati come segue:

Valore	[SHIFT] PREMUTO	[CTRL] PREMUTO	TASTO PREMUTO
–1	?	?	NO
0	NO	NO	SI
32	SI	NO	SI
128	NO	SI	SI
130	SI	SI	SI

Istruzioni associate: INPUT, INKEY$

INKEY$

```
10 CLS
20 PRINT "Sei bravo (s/n) ?"
30 A$=INKEY$:IF A$=" " THEN 30
40 IF A$="N" OR A$="n" THEN PRINT "...modesto...":GOTO 10
50 IF A$="S" OR A$="s" THEN PRINT "...come me...":GOTO 10
```

FUNZIONE: Legge la tastiera permettendo all'operatore l'interattività con il computer senza dover batter [ENTER] dopo ogni risposta. Nel caso che non sia premuto alcun tasto viene restituita una stringa vuota " ".

Istruzioni associate: INPUT, INKEY

INP

INP (<numero di porta>)

PRINT INP(&FF77)

FUNZIONE: Restituisce il dato presente in ingresso sulla porta di I/O mappata in memoria all'indirizzo specificato.

Istruzioni associate: OUT, WAIT

INPUT

INPUT [#<canale>,][;][<stringa>;]<elenco di [variabili]>

```
10 CLS
20 INPUT "Dammi due numeri, separati da una virgola ";A,B
30 IF A=B THEN PRINT "I due numeri sono uguali"
40 IF A>B THEN PRINT A " è maggiore di " B
50 IF A<B THEN PRINT A " è minore di " B
60 CLEAR:GOTO 20
```

COMANDO: Legge dati dal <canale> stabilito. Un punto e virgola [;] dopo la INPUT evita che il cursore passi a una nuova riga alla fine della risposta. Un punto e virgola dopo la <stringa> causa la stampa di un punto interrogativo, mentre una virgola sopprime il punto interrogativo. Nel caso si verificassero errori durante una INPUT il BASIC risponderebbe con:

?Redo from start   seguito dalla ripetizione della richiesta.

Tutte le risposte alla INPUT devono terminare con un [ENTER]. Il punto e virgola dopo il <canale> impedisce al cursore di tornare a capo dopo la risposta all'INPUT. Nel caso in cui venga indicato come <canale> un file su cassetta, la INPUT non genera alcun messaggio, ignorando anche quello eventualmente presente tra doppi apici.

Per ogni variabile presente nella lista viene letto un elemento del file. L'elemento letto deve essere compatibile con il tipo di variabile specificato nell'istruzione INPUT; deve trattarsi, cioè, di un dato numerico seguito da una virgola, un return, uno spazio o dall'EOF. Le virgole o gli [ENTER] posti dopo lo spazio separatore sono ignorati. Le stringhe racchiuse tra doppi apici vengono lette così come sono. La fine delle stringhe non racchiuse tra doppi apici è indicata nello stesso modo che per i dati numerici.

Istruzioni associate: LINE INPUT, READ, INKEY$

INSTR

INSTR ([<espressione intera>,]<stringa>,<stringa>)

PRINT INSTR(2, "BANANA","AN")
 2

FUNZIONE: Esamina la prima <stringa> per determinare la posizione in cui, eventualmente, compare la seconda <stringa>.

Il parametro opzionale <espressione intera> indica il punto di inizio della ricerca nella prima <stringa>; in mancanza di tale parametro la ricerca parte dal primo carattere.

Istruzioni associate: MID$, LEFT$, RIGHT$

INT

INT (<espressione numerica>)

PRINT INT(−1.995)
−2

FUNZIONE: Arrotonda per difetto l'<espressione numerica> all'intero più prossimo. Pertanto il comportamento della INT sui numeri positivi è identico a quelo della FIX, mentre sui numeri negativi non interi il risultato restituito dalla INT è sempre minore di una unità rispetto a quello della FIX.

Istruzioni associate: CINT, FIX, ROUND

JOY

JOY (<espressione intera>)
10 IF JOY(0)=8 THEN GOTO 100

FUNZIONE: Restituisce la posizione in cui si trova il joystick specificato da <espressione intera> (0 o 1).

POSIZIONE	VALORE CORRISPONDENTE
in alto	1
in basso	2
a sinistra	4
a destra	8
fuoco 2	16
fuoco 1	32

Istruzioni associate: INKEY

KEY

KEY <espressione intera>,[CHR$(n)+]<stringa>[+CHR$(n)]

KEY 140, "RUN"+CHR$(13)

COMANDO: Assegna al tasto dichiarato nella <espressione intera> (valore compreso tra 128 e 159. Vedi Appendice III) una stringa di lunghezza massima pari a 32 caratteri. La stringa associata può essere un comando del BASIC, come nell'esempio, seguito da un RETURN che lo rende eseguibile. La somma totale della lunghezza di tutte le stringhe associate ai tasti ridefiniti non può superare 120.

Istruzioni associate: KEY DEF

KEY DEF

KEY DEF <cod. tasto>, <repeat>[,<normale>[,<+SHIFT>[,<+CTRL>]]]

KEY DEF 46,1,63

COMANDO: Converte il codice generato da ogni tasto (come definito nell'Appendice III) nel nuovo assegnatogli. L'esempio fa in modo che la pressione del tasto 'n' corrisponda alla stampa di un punto interrogativo [?], avendo praticamente convertito il codice di tastiera 46 (n) in codice ASCII 63 (?).

Per riportare la situazione alla normalità:

KEY DEF 46,1,110

dove il carattere di codice ASCII 110 rappresenta la lettera 'n' minuscola.

Istruzioni associate: KEY

LEFT$

LEFT$(<stringa>,<espressione intera>)

```
10 CLS
20 A$ = "AMSTRAD"
30 B$ = LEFT$(A$,3)
40 PRINT B$
```

```
RUN

AMS
Ready
```

FUNZIONE: Estrae dalla <stringa> il numero di caratteri specificato dalla <espressione intera> partendo da sinistra. Nel caso in cui la lunghezza specificata in <espressione intera> sia superiore a quella della <stringa>, quest'ultima verrà restituita integralmente.

Istruzioni associate: MID$, RIGHT$

LEN

LEN (<stringa>)

A$ = "AMSTRAD" : PRINT LEN (A$)

FUNZIONE: Restituisce il numero di caratteri di cui è composta la <stringa>, spazi compresi.

LET

LET <variabile> = <espressione>

LET x = 100

COMANDO: Assegna alla <variabile> specificata il valore della <espressione>posta a destra del segno uguale. Nel BASIC AMSTRAD può essere omessa e serve solo per conservare una certa compatibilità con BASIC meno evoluti. L'esempio può essere abbreviato in:

x = 100

LINE INPUT

LINE INPUT [#<canale>,][;][<stringa>;]<variabile stringa>

LINE INPUT A$

LINE INPUT "NOME";N$

COMANDO: Legge un'intera riga dal <canale> indicato. Nel caso il <canale> non risulti specificato, il BASIC legge automaticamente dal canale 0, cioè la console. Inserendo un punto e virgola dopo LINE INPUT si evita di fare tornare il cursore a capo.

Istruzioni associate: READ, INPUT, INKEY$, INPUT$

LIST

LIST [<numero di riga>][–][<numero di riga>][,#<canale>]

LIST 100–1000,#1

COMANDO: Invia sul <canale> scelto il listato delle righe di programma comprese fra gli estremi specificati. Il listing può essere sospeso premendo [ESC] e ripreso con la pressione della barra spaziatrice. Battendo per due volte in sequenza [ESC] si ritorna al modo diretto del BASIC. Il listing può essere richiesto dall'inizio del programma sino a un dato punto, o da una determinata riga sino al termine. Ad esempio:

LIST –200   oppure   LIST 30–

LOAD

LOAD <nome file>[,<indirizzo>]

LOAD "INVENT"

COMANDO: Carica un programma BASIC da cassetta in memoria, sostituendo ogni programma eventualmente presente. Utilizzando la parte opzionale <indirizzo> è possibile caricare file binari, come spiegato nel Capitolo 2.

LOCATE

LOCATE [#<canale>,]<coordinata x>,<coordinata y>

```
10 MODE 1
20 LOCATE 20,12
30 PRINT CHR$(249)
```

COMANDO: Posiziona il cursore alle coordinate specificate, relative all'origine della WINDOW. I valori di default definiscono come 'finestra' l'intero schermo con origine nell'angolo in alto a sinistra, di coordinate 1,1.

Istruzioni associate: WINDOW

LOG

LOG (<espressione>)

```
?LOG (9999)
 9.21024037
```

FUNZIONE: Calcola il logaritmo naturale della <espressione>; restituisce un numero reale anche se l'<espressione> può essere in notazione intera.

Istruzioni associate: EXP, LOG10

LOG10

LOG10 (<espressione>)

```
?LOG10 (9999)
 3.99995657
```

FUNZIONE: Calcola il logaritmo in base 10 della <espressione> e restituisce come risultato un numero reale, anche se l'<espressione> è in notazione intera.

Istruzioni associate: EXP, LOG

LOWER$

LOWER$ (<stringa>)

```
A$ = "AMSTRAD": PRINT LOWER$ (A$)
amstrad
```

FUNZIONE: Converte ogni carattere maiuscolo presente nella <stringa> nel corrispondente minuscolo. È utile per elaborare degli INPUT per i quali la risposta può giungere indifferentemente in maiuscolo o in minuscolo.

Istruzioni associate: UPPER$

MAX

MAX (<elenco di <espressioni>>)

```
10 n = 66
20 PRINT MAX (1,n,3,6,4,3)

RUN
 66
Ready
```

FUNZIONE: Trova l'<espressione> di valore maggiore fra quelle elencate.

Istruzioni associate: MIN

MEMORY

MEMORY <indirizzo>

```
MEMORY &20AA
```

COMANDO: Definisce l'<indirizzo> massimo di memoria accessibile al BASIC. Per verificare la quantità di memoria disponibile utilizzare il comando FRE.

Istruzioni associate: HIMEM, FRE

MERGE

MERGE [<nome file>]

MERGE "PLAN"

COMANDO: Concatena un programma proveniente da cassetta con quello presente in memoria. Se il nome del file non viene dichiarato il BASIC cerca di concatenare il primo programma incontrato sulla cassetta. Se il primo carattere del <nome file> è '!', allora tutti gli usuali messaggi visualizzati durante il caricamento da cassetta vengono soppressi.

Se due programmi che vengono concatenati contengono righe con lo stesso numero, quelle del programma presente in memoria risultano cancellate e sostituite da quelle del programma caricato da cassetta. Tutti i file aperti vengono abbandonati, così come le variabili e le funzioni definite dall'utente. Non è possibile eseguire il concatenamento di programmi protetti.

Istruzioni associate: LOAD, CHAIN, CHAIN MERGE

MID$

MID$ (<stringa>, <espressione intera> [, <espressione intera>])

A$ = "AMSTRAD": PRINT MID$ (A$,2,4)
MSTR

A$ = "AMSTRAD": B$=MID$ (A$,2,4): PRINT B$
MSTR

COMANDO E FUNZIONE: Definisce una substringa compresa nei limiti specificati dalle due <espressioni intere>. Tale substringa può essere poi assegnata ad una variabile o utilizzata all'interno di una espressione. La prima <espressione intera> specifica la posizione del primo carattere della substringa; la seconda <espressione intera> ne dichiara la lunghezza. Omettendo tale parametro il BASIC assume automaticamente la lunghezza della stringa originale.

Istruzioni associate: LEFT$, RIGHT$

MIN

MIN (<elenco di <espressioni>>)

PRINT MIN (3,6,2.999,8,9)
 2.999

FUNZIONE: L'opposto di MAX.

Istruzioni associate: MAX.

MODE

MODE <espressione intera>

MODE 1

COMANDO: È utilizzato per specificare il modo di organizzazione dello schermo. Esegue automaticamente la pulizia dello schermo utilizzando INK 0, a prescindere dall'attuale colore di fondo. Tutte le 'finestre' eventualmente aperte vengono chiuse e i cursori, sia di testo che grafico, riportati alle rispettive coordinate di origine.

Istruzioni associate: WINDOW, ORIGIN

MOVE

MOVE <coordinata x>, <coordinata y>

MOVE 34,34

COMANDO: Sposta il cursore grafico posizionandolo alle coordinate assolute specificate. Le FUNZIONI XPOS e YPOS possono essere utilizzate per stabilire l'attuale posizione del cursore grafico.

Istruzioni associate: MOVER, PLOT, PLOTR, DRAW, DRAWR, ORIGIN, TEST, TESTR, XPOS, YPOS.

MOVER

MOVER <spostamento sulle x>, <spostamento sulle y>

MOVER 34,34

COMANDO: Fa compiere al cursore grafico gli <spostamenti> specificati sulle X e sulle Y, partendo dalla posizione attuale.

Istruzioni associate: MOVE, PLOT, PLOTR, DRAW, DRAWR, TEST, TESTR, XPOS, YPOS

NEW

NEW

NEW

COMANDO: Cancella il programma presente in memoria. Le definizioni dei tasti attuate tramite il comando KEY non vengono alterate e lo schermo non viene

pulito. Utilizzare con cautela, dato che si tratta di un comando molto simile al RESET.

NEXT

NEXT [<elenco di <variabili>>]

FOR n=1 to 1000:NEXT

COMANDO: Chiude un ciclo aperto con il comando FOR. Il comando NEXT può essere privo di argomenti o riferirsi ad una specifica variabile. L'esempio diverrebbe:

NEXT n

Istruzioni associate: FOR

ON GOSUB - ON GOTO

ON <espressione intera> GOSUB <elenco di <numeri di riga>>

ON <espressione intera> GOTO <elenco di <numeri di riga>>

10 ON GIORNO GOSUB 100,200,300,400,500

10 ON RATE GOTO 1000,2000,3000,4000

COMANDO: L'esecuzione del programma riprende da una delle righe specificate, a seconda del valore della <espressione intera>. Nell'esempio, per GIORNO = 1 il programma salta alla subroutine che inizia alla riga 100, per GIORNO = 2 viene eseguita la subroutine che inizia alla riga 200 e così via.

Istruzioni associate: GOTO, GOSUB

ON BREAK GOSUB

ON BREAK GOSUB <numero di riga>

```
10 ON BREAK GOSUB 40
20 PRINT "Programma in esecuzione"
30 GOTO 20
40 CLS
50 PRINT "Premendo [ESC] due volte si esegue questa subroutine"
60 FOR t=1 TO 2000:NEXT
70 RETURN
```

COMANDO: A seguito dell'interrupt provocato dalla duplice pressione del tasto [ESC], viene richiamata la subroutine specificata.

Istruzioni associate: ON BREAK STOP, RETURN

ON BREAK STOP

ON BREAK STOP

```
10 ON BREAK GOSUB 40
20 PRINT "Programma in esecuzione"
30 GOTO 20
40 CLS
50 PRINT "Premendo due volte [ESC] si esegue questa subroutine"
60 FOR t=1 TO 2000:NEXT
70 ON BREAK STOP
80 RETURN
```

COMANDO: Quando inserito all'interno di una subroutine chiamata tramite una ON BREAK GOSUB, questo comando disabilita l'opzione ON BREAK GOSUB, ma non ha altri effetti immediati. Nel programma precedente, che include la ON BREAK STOP, l'opzione ON BREAK GOSUB opererà solo la prima volta.

Istruzioni associate: ON BREAK GOSUB

ON ERROR GOTO

ON ERROR GOTO <numero di riga>

```
10 ON ERROR GOTO 80
20 CLS
30 PRINT "Nel caso che il programma contenga un errore,"
40 PRINT "può essere comodo listare il programma"
50 PRINT "in modo da individuare l'errore"
60 FOR t=1 TO 4000:NEXT
70 GOTO 200:REM errore
80 CLS:PRINT "C'è un errore alla riga"; ERL:PRINT
90 LIST
```

COMANDO: Salta ad una riga specificata nel caso in cui si verifichi un errore durante l'esecuzione del programma.

Istruzioni associate: ERR, ERL, RESUME

ON SQ GOSUB

ON SQ (<canale>) GOSUB <numero di riga>
ON SQ (2) GOSUB 2000

COMANDO: Abilita un interrupt in attesa che la coda sul <canale> sonoro specificato presenti un posto libero. Il parametro <canale> può assumere uno dei seguenti valori:

1 per il canale A
2 per il canale B
4 per il canale C

Istruzioni associate: SOUND, SQ

OPENIN

OPENIN <nome file>

100 OPENIN "!INFORMAZIONI"

COMANDO: Apre in input un file da cassetta. Il file così aperto può essere letto e fornire dati al programma corrente.
Se il primo carattere del <nome file> è '!', vengono soppressi tutti gli usuali messaggi di caricamento.

Istruzioni associate: CLOSEIN, OPENOUT

OPENOUT

OPENOUT <nome file>

OPENOUT "!DATI"

COMANDO: Apre in output un file verso la cassetta. Al file, creato con il nome specificato, viene associato un buffer di 2k che è riempito dai vari blocchi in successione.

Nessuna informazione viene fisicamente scritta sul nastro sino a quando il buffer non risulta pieno o è eseguito un CLOSEOUT.
N.B. Il comando NEW provoca l'abbandono di tutti i file aperti e la perdita dei dati eventualmente contenuti nel buffer.

Istruzioni associate: CLOSEOUT, OPENIN

ORIGIN

ORIGIN<x>,<y>[,<lim. destro>, <lim. sinistro>,<lim. sup.>,<lim. inf.>]

```
10 CLS:BORDER 13
20 ORIGIN 0,0,50,590,350,50
30 DRAW 540,350
40 GOTO 20
```

COMANDO: Determina le coordinate iniziali del cursore grafico. La parte opzionale — [...] — contiene i parametri per la definizione di una nuova 'finestra' grafica, che opererà in qualunque Modo di visualizzazione.

Il sistema di riferimento sugli assi cartesiani ha origine nel punto 0,0 fissato nell'angolo in basso a sinistra dello schermo. Le nuove coordinate specificate per l'origine hanno sempre per riferimento questo punto.

Nel caso che la definizione della finestra contenga parametri incompatibili con la dimensione dello schermo, viene automaticamente assunta la massima dimensione possibile.

Istruzioni associate: WINDOW.

OUT

OUT <numero di porta>, <espressione intera>

```
OUT &F8F4,10
```

COMANDO: Invia il valore della <espressione intera>, che deve essere compreso fra 0 e 255, sulla porta il cui indirizzo è specificato tramite il <numero di porta>.

Istruzioni associate: INP, WAIT

PAPER

PAPER [#<canale>,]<colore>

```
10 MODE 0
20 FOR p=0 TO 15
30 PAPER p:CLS
40 PEN 15-p
50 LOCATE 6,12:PRINT "SFONDO"p
60 FOR t=1 to 500: NEXT t
70 NEXT p
```

COMANDO: Seleziona il colore di fondo. Prima di procedere alla stampa su schermo, la matrice del carattere assume il colore di fondo, a meno che non risulti selezionato il modo trasparente.

Il numero di colori visualizzabili contemporaneamente dipende dal Modo selezionato. Se il <colore> specificato non è disponibile, al fondo verrà assegnato per default un colore tra quelli disponibili.

Istruzioni associate: INK, WINDOW, PEN

PEEK

PEEK (<indirizzo>)

```
10 MODE 2
20 INK 1,0:INK 0,12: BORDER 12
30 INPUT "Indirizzo iniziale"; primo
40 INPUT "Indirizzo finale"; ultimo
50 FOR i=primo TO ultimo
60 v$ = HEX$ (PEEK(i),2)
70 PRINT v$;
80 PRINT "alla locazione";HEX$(i,4),
90 NEXT
```

FUNZIONE: Esamina il contenuto della locazione di memoria specificata da <indirizzo>. Il programma d'esempio permette di 'vedere' tutta la RAM del CPC 464.

Istruzioni associate: POKE

PEN

PEN [#<canale>,]<colore>

PEN 1,2

COMANDO: La PEN seleziona il <colore> che deve essere utilizzato per scrivere sul <canale> video specificato, che è per default il numero 0.

Istruzioni associate: INK, PAPER

PI

PI

```
PRINT PI
  3.14159265
```

```
 20 MODE 2
 30 RAD
 40 INK 1,0
 50 INK 0,12
 60 BORDER 9
 70 FOR N=1 TO 200
 80 ORIGIN 420,0
 90 DRAW 0,200
 95 REM traccia gli angoli dal punto di fuga
100 DRAW 30*N*SIN(N*PI/4),(SIN(PI/2))*N*SIN(N)
110 NEXT
```

```
120 MOVE 0,200
130 DRAWR 0,50
140 DRAWR 40,0
150 WINDOW 1,40,1,10
```

FUNZIONE: È il valore del rapporto fra la circonferenza e il diametro di un cerchio. Viene spesso utilizzata nelle routine grafiche, come quella d'esempio.

Istruzioni associate: DEG, RAD

PLOT

PLOT <coordinata x>,<coordinata y>[,<colore>]

```
10 MODE 2:PRINT "Inserisci 4 numeri, separati da virgole":PRINT
20 PRINT "Inserisci l'origine delle X (0—639) e quella delle Y (0-399), il
raggio e l'angolo da coprire":INPUT x,y,r,s
30 ORIGIN x,y
40 FOR angolo=1 TO 360 STEP S
50 PUNTOX = r*COS(angolo)
60 PUNTOY = r*SIN(angolo)
70 PLOT PUNTOX,PUNTOY
74 REM MOVE 0,0
75 REM DRAW PUNTOX,PUNTOY
80 NEXT
```

COMANDO: Provate come prima risposta 320,200,20,1. La PLOT è uguale alla DRAW, tranne che solo il pixel finale viene acceso. Se togliete il REM dalla riga 75 e ne inserite uno all'inizio della 70 per renderla inoperante, vi accorgerete della differenza. Togliete il REM dalla riga 74 per riempire il cerchio.

Notate che il processo di riempimento del cerchio è ottenuto tramite il disegno di circonferenze concentriche. Ricordate che in questo programma gli angoli sono misurati in radianti. Per passare ai gradi inserite la riga:

```
25 DEG
```

e ridate RUN.

Istruzioni associate: DRAW, DRAWR, PLOTR, MOVE, MOVER, ORIGIN, TEST, TESTR, XPOS, YPOS

PLOTR

PLOTR <coordinata x>,<coordinata y>[,<colore>]

```
10 MODE 2:PRINT "Inserisci 4 numeri, separati da virgole":PRINT
```

```
20 PRINT "Inserisci l'origine delle X (0-639) e quella delle Y (0-399), il
raggio e l'angolo da coprire":INPUT x,y,r,s
30 ORIGIN x,y
40 FOR angolo=1 TO 360 STEP s
50 PUNTOX = r*COS(angolo)
60 PUNTOY = r*SIN(angolo)
70 PLOTR PUNTOX, PUNTOY
80 NEXT:GOTO 40
```

COMANDO: Provate come prima risposta 320,0,20,1. La PLOTR è uguale alla DRAWR, tranne che solo il pixel finale viene acceso.

Istruzioni associate: DRAW, DRAWR, PLOT, MOVE, MOVER, ORIGIN, TEST, TESTR, XPOS, YPOS

POKE

POKE <indirizzo>,<espressione intera>

```
POKE &00FF,10
```

COMANDO: Pone nella locazione di memoria specificata il valore della <espressione intera>. Deve essere utilizzata con cautela perché può facilmente provocare il crash del sistema.

Istruzioni associate: PEEK

POS

POS (#<canale>)

```
PRINT POS (#0)
 1
```

FUNZIONE: Restituisce la posizione attuale del cursore. È obbligatorio specificare il parametro <canale>, pena il messaggio Syntax error.

Se il <canale> è lo schermo, la funzione restituisce l'attuale posizione sull'asse delle X del cursore; tale posizione è relativa all'origine della 'finestra', dove l'angolo in alto a sinistra ha coordinate 1,1.

Se il <canale> è la stampante, la funzione restituisce la posizione del carrello, assegnando il valore 1 al margine sinistro. Dal conteggio sono esclusi i caratteri di controllo.

Istruzioni associate: VPOS

PRINT

PRINT [#<canale>,][<argomenti>][<USING>][<separatore>]]

PRINT #0, "a b c"

COMANDO: Vedere alla fine di questo capitolo.
Istruzioni associate: USING, TAB, SPC

RAD

RAD

RAD

COMANDO: Seleziona il RADIANTE come unità di misurazione delle ampiezze angolari.

Istruzioni associate: DEG, SIN, COS, TAN, ATN

RANDOMIZE

RANDOMIZE [<espressione numerica>]

```
10 RANDOMIZE 23
20 PRINT RND (6)
```

COMANDO: Il generatore di numeri casuali del BASIC produce una sequenza di numeri pseudo casuali dove ognuno dipende dal precedente. Partendo da un valore dato la sequenza è sempre la stessa. Usando il comando RANDOMIZE si ottiene la generazione di un nuovo valore iniziale da passare al generatore di numeri casuali. La forma RANDOMIZE TIME genera una serie di difficile ripetizione.

Istruzioni associate: RND

READ

READ <elenco di variabili>

```
10 FOR X=1 TO 3
20 READ N$
30 PRINT N$
40 DATA PAOLO,MARIA, ANNA
50 NEXT
```

COMANDO: Legge le costanti definite negli statement DATA assegnandole alle <variabili>. La scansione della lista di costanti è automatica. Il comando RESTORE riposiziona il puntatore utilizzato dalla READ all'inizio dello statement DATA.

Istruzioni associate: DATA, RESTORE

RELEASE

RELEASE <canale sonoro>

RELEASE 4

COMANDO: Viene utilizzato per togliere dallo stato d'attesa eventuali suoni in coda sul <canale sonoro>. L'espressione che identifica il <canale> deve essere così codificata:

canale A : 1
canale B : 2
canale C : 4

Perciò il valore 4 dell'esempio toglie dallo stato d'attesa i suoni in coda sul canale C.

Istruzioni associate: SOUND

REM

REM <resto della riga>

10 REM copyright 1984 by MICROSTAR

COMANDO: Permette di inserire commenti nel listato del programma BASIC. Un singolo apice (') equivale a :REM, ad eccezione che nei comandi DATA dove viene considerato come parte di una stringa non racchiusa tra doppi apici.

REMAIN

REMAIN (<espressione intera>)

REMAIN (3)
PRINT#6,REMAIN(0);

FUNZIONE: Disabilita il timer specificato (da 0 a 3), dopo aver letto il valore del tempo rimanente sul timer. Restituisce 0 se il timer non risulta abilitato.

Istruzioni associate: AFTER, EVERY

RENUM

RENUM [<nuovo num. di riga>][,<vecchio num. di riga>][,<passo>]

RENUM 100,,100

COMANDO: Rinumera le righe di programma incluse fra gli estremi specificati, utilizzando il <passo> definito. Il <nuovo num. di riga> da' il primo numero della nuova sequenza, pari a 10 per default. Il <vecchio num. di riga> indica la riga da cui iniziare la rinumerazione: se omesso il RENUM parte dalla prima riga del programma. Il <passo> indica l'incremento fra un numero di riga e l'altro, anch'esso pari a 10 per default.

I numeri di riga validi sono quelli compresi fra 1 e 65535.

RESTORE

RESTORE [<numero di riga>]

RESTORE 300

```
10 FOR N=1 TO 6
20 READ A$
30 PRINT A$ " ";
40 DATA Dopo la, RESTORE i, dati possono, essere, nuovamente, letti
50 NEXT
60 PRINT
70 RESTORE
80 GOTO 10
```

COMANDO: Riposiziona il puntatore di lettura all'inizio del DATA presente al <numero di riga> specificato. Nel caso questa opzione risulti omessa, il comando agisce sul primo DATA incontrato.

Istruzioni associate: READ, DATA

RESUME

RESUME [<numero di riga>]

oppure

RESUME NEXT

RESUME 300

COMANDO: Dopo l'individuazione di un errore tramite un comando ON ERROR GOTO, la RESUME permette la ripresa dell'esecuzione dalla riga specificata. La forma RESUME riprende dalla riga in cui si è verificato l'errore, mentre la forma RESUME NEXT riprende dalla riga immediatamente successiva a questa.

Istruzioni associate: ON ERROR GOTO

RETURN

RETURN

RETURN

COMANDO: Indica il termine di un sottoprogramma e continua l'esecuzione del programma dall'istruzione successiva al comando GOSUB chiamante.

Istruzioni associate: GOSUB, ON x GOSUB, ON SQ GOSUB, AFTER n GOSUB, EVERY n GOSUB, ON BREAK GOSUB

RIGT$

RIGHT$ (<stringa>,<espressione intera>)

```
10 CLS
20 A$ = "AMSTRAD"
30 B$ = RIGHT$ (A$,3)
40 PRINT B$
RUN
```

```
RAD
Ready
```

FUNZIONE: Estrae dalla <stringa> un gruppo di caratteri, di lunghezza pari a <espressione intera>, partendo da destra. Se la <stringa> è più corta di quanto richiesto, la funzione restituisce l'intera <stringa>.

Istruzioni associate: MID$, LEFT$

RND

RND [<espressione>]

```
10 RANDOMIZE 23
20 PRINT RND (6)
```

FUNZIONE: Genera un numero casuale. Il comando RANDOMIZE dell'esempio assicura che il numero generato sia sempre lo stesso (0.146940658).

RND(0) restituisce la copia dell'ultimo numero generato.

Istruzioni associate: RANDOMIZE

ROUND

ROUND (<espressione>,<espressione intera>)

```
10 x = 0.123456789
20 FOR r=9 TO 0 STEP -1:PRINT r,ROUND(x,r):NEXT
30 x = 123456789
40 FOR r=0 TO -9 STEP -1
50 PRINT r,ROUND(x,r)
60 NEXT
```

FUNZIONE: Arrotonda l'<espressione> conservando il numero di cifre decimali specificato dalla <espressione intera>. Se quest'ultima è minore di 0, l'<espressione> viene arrotondata in modo da presentare prima della virgola decimale un numero di zeri pari al valore assoluto di <espressione intera>.

Istruzioni associate: INT, FIX, CINT, ABS

RUN

RUN "<nome file>"

RUN "WELCOME"

COMANDO: Carica un programma da cassetta e lo esegue. Se il <nome file> è una <stringa> vuota (" ") il BASIC cerca di portare in memoria ed eseguire il primo programma incontrato sul nastro. Se il primo carattere della <nome file> è '!', tutti gli usuali messaggi visualizzati durante il caricamento vengono soppressi.

Istruzioni associate: LOAD

RUN

RUN [<numero di riga>]

RUN 100

COMANDO: Inizia l'esecuzione del programma presente in memoria dalla riga specificata, o dall'inizio se tale parametro viene omesso. Le variabili e le funzioni definite dall'utente vengono azzerate e tutti i file abbandonati.

Istruzioni associate: LOAD

SAVE

SAVE <nome file>[,<tipo file>][,<parametri per file binari>]

SAVE "PROG",P

COMANDO: Salva sulla cassetta il programma presente in memoria, con il <nome file> specificato.

Opzioni:

,A salva un file in formato ASCII
,P protegge il file
,B salva un'area di memoria come file binario: ad es. lo schermo

Istruzioni associate: LOAD, MERGE, CHAIN, CHAIN MERGE, RUN "<nome file>"

SGN

SGN (<espressione>)

```
10 INPUT "Come sta il tuo conto in banca"; conto
20 IF SGN (conto) < 1 GOTO 30 ELSE 40
30 PRINT "Sei conciato male !!": END
40 PRINT "Sei più ricco di me"
```

FUNZIONE: Determina il segno della <espressione>. Restituisce −1 se l'<espressione> è minore di 0, oppure 0 se l'<espressione> è uguale a 0. Altrimenti restituisce 1.

Istruzioni associate: ABS

SIN

SIN (<espressione>)

```
PRINT SIN (PI/2)
 1
```

FUNZIONE: Calcola il valore Reale del Seno della <espressione>, tenendo conto dell'unità di misurazione angolare in uso (radianti per default).

Istruzioni associate: COS, TAN, ATN, DEG, RAD

SOUND <stato del canale sonoro>,<periodo del tono>[,<durata>[,<volume>[,<inviluppo di volume>[,<inviluppo di tono>[,<periodo del rumore>]]]]]

SOUND 1,200,1000,7,0,0,1

COMANDO: Le possibilità sonore del CPC 464 sono molto sofisticate: si consiglia la lettura del Capitolo 6.

Istruzioni associate: ENV, ENT

SPACE$

SPACE$ (<espressione intera>)

SPACE$ (190)

FUNZIONE: Crea una stringa di spazi di lunghezza pari a <espressione intera>.

Istruzioni associate: PRINT, SPC, TAB

SPEED INK

SPEED INK <espressione intera>,<espressione intera>

```
10 INK 0,9,12:INK 1,0,26
20 BORDER 12,9
30 SPEED INK 50,20
```

COMANDO: I comandi INK e BORDER permettono di associare due colori ad ogni inchiostro; in tal modo l'inchiostro si alterna tra i due colori selezionati. Le due <espressioni intere> specificano i tempi di utilizzo rispettivamente del primo e del secondo colore. Ricordate che tutti i tempi sono misurati in cinquantesimi di secondo.

Istruzioni associate: INK, BORDER

SPEED KEY

SPEED KEY <ritardo iniziale>,<tempo di ripetizione>

SPEED KEY 20,3

COMANDO: Serve a variare i parametri per la ripetizione automatica dei tasti dopo il <ritardo iniziale> e con la frequenza data dal <tempo di ripetizione>. I valori di default corrispondono ad uno SPEED KEY 10,10; i tempi sono sempre in cinquantesimi di secondo.

Istruzioni associate: KEY DEF

SPEED WRITE

SPEED WRITE <espressione intera>

SPEED WRITE 1

COMANDO: Dati e programmi possono essere salvati su cassetta sia alla velocità di 2000 baud (quando <espressione intera>=1), che a quella standard di 1000 baud (quando <espressione intera>=0).

Durante il caricamento del file il CPC 464 stabilisce automaticamente la corretta velocità di lettura della cassetta, che non deve, quindi, essere specificata.

Con cassette di scarsa qualità è consigliabile utilizzare solo la velocità di scrittura standard, che risulta più affidabile. Vedere il Capitolo 2 per ulteriori informazioni.

Istruzioni associate: SAVE

SQ

SQ (<canale sonoro>)

```
10 MODE 1
20 FOR n=20 TO 0 STEP -1
30 PRINT n;
40 SOUND1,10+n,100,7
50 WHILE SQ(1) > 127:WEND
60 NEXT
```

FUNZIONE: È utilizzata per controllare il numero di posti liberi sul <canale sonoro> specificato, dove il canale A ha valore 1, il B ha valore 2 ed il C ha valore 4. La funzione determina anche se il canale è attivo e, in caso contrario, il motivo per cui l'eventuale posto all'inizio della coda è in attesa. Il risultato restituito deve essere così interpretato:

bit 0,1 e 2	indicano il numero di posti liberi;
bit 3,4 e 5	indicano l'eventuale stato di Appuntamento all'inizio della coda;
bit 6	è settato se il principio della coda è in stato di Attesa;
bit 7	è settato se il canale è attivo

Istruzioni associate: SOUND, ON SQ GOSUB

SQR

SQR (<espressione>)

```
PRINT SQR (9)
 3
```

FUNZIONE: Calcola la radice quadrata della <espressione>.

Istruzioni associate: PRINT

STOP

```
STOP

300 IF n<0 THEN STOP
```

COMANDO: Interrompe l'esecuzione del programma, pur lasciando il BASIC in uno stato per cui la stessa può riprendere. Può essere utilizzato durante la fase di sviluppo per fermare il programma in un punto determinato o al verificarsi di una particolare condizione.

Istruzioni associate: CONT, END

STR$

```
STR$(<espressione>)

PRINT STR$(&766)
 1894

PRINT STR$(&X1010100)
 84
```

FUNZIONE: Converte l'<espressione> nella sua rappresentazione decimale sotto forma di stringa. Utilizza lo stesso formato della PRINT.

Istruzioni associate: VAL, PRINT, HEX$, BIN$

STRING$

```
STRING$(<espressione intera>,<carattere>)

PRINT STRING$(&16,"*")
**********************
```

FUNZIONE: Crea una stringa costituita dal <carattere> ripetuto tante volte quante specificate nella <espressione intera>.

SYMBOL

```
SYMBOL <codice carattere>,<definizione matrice del carattere>
```

```
10 MODE 2
30 SYMBOL 241,&80,&40,&20,&10,&8,&4,&2,&1
40 FOR n=1 TO 2000
50 PRINT CHR$(241);
60 NEXT
70 GOTO 70
```

COMANDO: Ridefinisce la matrice del carattere di codice specificato. Il codice carattere deve essere normalmente maggiore di 239, a meno di non usare il comando SYMBOL AFTER. La matrice 8×8 del carattere selezionato viene ridefinita riga per riga, a partire da quella superiore, secondo quanto indicato dai parametri di <definizione>. La codifica binaria di ciascuno dei parametri determina il colore con cui vengono accesi i pixel sulla singola riga della matrice: colore di fondo per i bit posti a 0 e colore dell'INK corrente per i bit settati a 1.

Il programma di esempio ridefinisce la matrice del carattere di codice 241 ottenendo il simbolo '\'.

Istruzioni associate: SYMBOL AFTER

SYMBOL AFTER

SYMBOL AFTER <espressione intera>

SYMBOL AFTER 90

COMANDO: Tramite questo comando è possibile modificare il numero di caratteri ridefinibili dall'utente. Assegnando, per esempio, alla <espressione intera> il valore 32 si possono ridefinire tutti i caratteri il cui codice è compreso fra 32 e 255.

Ogni volta che viene eseguito un comando SYMBOL AFTER tutti i caratteri precedentemente ridefiniti sono riportati allo stato iniziale. Mancando questo comando il CPC 464 assume automaticamente come primo carattere ridefinibile quello di codice 240, lasciando all'utente la possibilità di ridefinire 16 caratteri.

Istruzioni associate: SYMBOL

TAG

TAG [#<canale>]

```
10 MODE 2
20 BORDER 9
30 INK 0,12
40 INK 1,0
50 FOR n=1 TO 100
```

```
 60 MOVE 200+n,320+n
 70 TAG
 80 IF n<70 GOTO 90 ELSE 100
 90 PRINT "Salve";: GOTO 110
100 PRINT "Addio";
110 NEXT
120 GOTO 50
```

COMANDO: Questo comando fa si che il testo inviato sul <canale> venga stampato a partire dalla posizione del cursore grafico, consentendo di miscelare testo e grafica. Il <canale> è per default lo 0. L'angolo in alto a sinistra della matrice del carattere da stampare viene posizionato sulle coordinate del cursore grafico. Vengono visualizzati anche i caratteri di controllo normalmente non stampabili, a meno che lo statement PRINT non sia seguito da un punto e virgola.

Istruzioni associate: TAGOFF

TAGOFF

TAGOFF [#<canale>]

TAGOFF #0

COMANDO: Disabilita il comando TAG. Dopo l'esecuzione del TAGOFF il testo viene inviato alla posizione in cui si trovava il cursore al momento del richiamo della TAG.

Istruzioni associate: TAG

TAN

TAN (<espressione>)

PRINT TAN (45)

FUNZIONE: Calcola la tangente dell'angolo dato dal valore della <espressione>, che deve essere compreso fra –200000 e +200000. Per default l'ampiezza dell'angolo viene misurata in radianti.

Istruzioni associate: COS, SIN, ATN, DEG, RAD

TEST

TEST (<coordinata x>,<coordinata y>)

PRINT TEST (300,300)

FUNZIONE: Restituisce il codice del colore che si trova alle coordinate grafiche specificate.

Istruzioni associate: TESTR, MOVE, MOVER, PLOT, PLOTR, DRAW, DRAWR

TESTR

TESTR (<spostamento sulle x>,<spostamento sulle y>)

TESTR (5,5)

FUNZIONE: Restituisce *il codice del colore che si trova alle* coordinate specificate, espresse in modo relativo rispetto alla posizione corrente del cursore grafico.

Istruzioni associate: TEST, MOVE, MOVER, PLOT, PLOTR, DRAW, DRAWR

TIME

TIME

```
10 DATO = INT(TIME/300)
20 TICTAC = ((TIME/300)-DATO)
30 PRINT TICTAC;
40 GOTO 20
```

FUNZIONE: Indica il tempo trascorso dall'accensione del CPC 464, escluso quello dedicato alle operazioni di lettura/scrittura da e verso cassetta. L'unità di tempo è il trecentesimo di secondo.

TRON - TROFF

TRON

TROFF

COMANDI: Il BASIC AMSTRAD permette di controllare l'esecuzione del programma, visualizzando tra parentesi quadre [] il numero della prossima riga da interpretare. Il comando TRON abilita questa possibilità, TROFF la disabilita.

Istruzioni associate: RUN

UNT

UNT (<indirizzo>)

```
PRINT UNT(&FF66)
-154
```

FUNZIONE: Converte un numero intero non segnato e compreso tra 0 e 65535 in intero con segno compreso fra –32768 e +32767.

Istruzioni associate: INT, FIX, CINT, ROUND

UPPER$

UPPER$(<stringa>)

PRINT UPPER$("amstrad")
AMSTRAD

FUNZIONE: Converte qualsiasi lettera minuscola presente nella <stringa> nell'equivalente maiuscola.

Istruzioni associate: LOWER$

VAL

VAL (<stringa>)

```
10 A$ ="7 è il mio numero fortunato"
20 PRINT VAL (A$)
```

FUNZIONE: Estrae una <espressione numerica> dalla <stringa>, a patto che detta <espressione numerica> si trovi all'inizio della <stringa>. L'opposto di STR$.

Istruzioni associate: STR$

VPOS

VPOS (#<canale>)

PRINT VPOS (#0)

FUNZIONE: Restituisce la posizione verticale del cursore per il <canale>. Come per la POS, non esiste un <canale> di default.

Istruzioni associate: POS

WAIT

WAIT <numero di porta>,<maschera>[,<inversione>]

WAIT &FF34,20,25

COMANDO: Sospende tutte le operazioni sino a quando la porta specificata non restituisce un valore particolare, compreso fra 0 e 255. Il BASIC lancia un loop di lettura sulla porta di I/0 ed esegue prima la XOR tra la <inversione> e i valori letti e successivamente la AND con la <maschera> fino a che il risultato è diverso da 0. Il BASIC può anche non uscire mai dal loop in quanto le condizioni richieste potrebbero non verificarsi. Il programma di esempio provoca il blocco totale del CPC 464, che dovrà essere totalmente resettato.

Istruzioni associate: INP, OUT

WEND

WEND

```
 10 MODE 1
 20 INPUT "inserisci l'ora (h,m,s)";h,m,s
 30 CLS: d = INT (TIME/300)
 40 WHILE h<13
 50 WHILE m<60
 60 WHILE t<60
 70 t = (INT (TIME/300) - d) + s
 80 LOCATE 70,4
 90 PRINT#0,USING "##";h,m,t
100 WEND
110 t = 0
120 s =0
130 m = m + 1
140 GOTO 30
150 WEND
160 m = 0
170 h = h + 1
180 WEND
190 h = 1
200 GOTO 40
```

COMANDO: Il ciclo WHILE/WEND esegue reiteratamente una parte di programma sino a quando la condizione posta nello WHILE risulta vera. Permette di strutturare i programmi rendendoli più leggibili e facilmente correggibili. Il comando WEND indica la fine di un ciclo WHILE, come il NEXT per i cicli FOR.

Istruzioni associate: WHILE

WHILE

WHILE <espressione logica>
WHILE GIORNO < 0

vedere l'esempio sopra

COMANDO: La WHILE indica l'inizio di un ciclo WHILE/WEND. La parte di programma compresa fra la WHILE e la WEND viene eseguita se la <espressione logica> è vera.

Istruzioni associate: WEND

WIDTH

WIDTH <espressione intera>
WIDTH 86

COMANDO: Indica al BASIC il numero di caratteri stampabili dalla stampante su ogni riga, permettendogli di inserire un carattere di ritorno carrello dove necessario.

Istruzioni associate: PRINT, POS

WINDOW

WINDOW [#<canale>,]<margine sinistro>,<margine destro>,<limite superiore>,<limite inferiore>

```
10 MODE 1
20 BORDER 6
30 WINDOW 10,30,7,18
40 PAPER 2:PEN 3
50 CLS
60 PRINT CHR$(143);CHR$(242);"Questa è la posizione"
70 PRINT "1,1 nella finestra di testo"
80 GOTO 80
```

COMANDO: Seleziona una finestra di testo per il <canale> definito. Il <canale> è lo 0 per default.

Istruzioni associate: ORIGIN

WINDOW SWAP

WINDOW SWAP <canale>, <canale>
WINDOW SWAP 0,2

COMANDO: Scambia fra loro due finestre di testo. Ad esempio, i messaggi del BASIC, normalmente inviati sul canale 0, possono essere diretti verso una nuova finestra per aiutare la programmazione.

Istruzioni associate: WINDOW, PEN, PAPER, TAG

WRITE

WRITE [#<canale>,] [<lista di espressioni>]

```
WRITE #2, "SALVE", 4,5
"SALVE",4,5
```

COMANDO: Stampa il valore di una <lista di espressioni> sul <canale> dato, comprese le virgole e i doppi apici.

Istruzioni associate: PRINT

XPOS

XPOS

```
PRINT XPOS
```

FUNZIONE: Restituisce la posizione orizzontale del cursore grafico.

Istruzioni associate: YPOS, MOVE, MOVER, ORIGIN

YPOS

YPOS

```
PRINT YPOS
```

FUNZIONE: Restituisce la posizione verticale del cursore grafico.

Istruzioni associate: XPOS, MOVE, MOVER, ORIGIN

ZONE

ZONE <espressione intera>

```
10 PRINT 1,2,3
20 ZONE 19
30 PRINT 4,5,6
```

COMANDO: Modifica l'ampiezza della Zona di Stampa della PRINT, che è 13 per default. L'effetto di questo comando è annullato dai comandi NEW, LOAD, CHAIN e RUN "nome file".

Istruzioni associate: PRINT, WIDTH

PRINT

PRINT [#<canale>,] [<elenco espressioni>] [<USING>] [<separatore>]

<elenco espressioni> è definito come:
<espressione> [<separatore><elemento da stampare>]

a sua volta <elemento da stampare> può essere:

<espressione>
SPC (<espressione intera>)
TAB (<espressione intera>)

Come scrivere dati in un file su cassetta:

```
10 OPENOUT "DATI"
20 PRINT#9, "SALVE"
30 CLOSEOUT
```

Come inviare caratteri ad una stampante:

```
10 STIPEN = 23000 * PI
20 PRINT #8,USING "#####.##"; STIPEN
30 PRINT #0,STIPEN
RUN
 72256.6311        [sullo schermo]
Ready

[intanto sulla stampante.....]
 72256.63
```

COMANDO: Invia i dati sul <canale> utilizzando il formato specificato. Vedere la tabella che segue per i caratteri di formattazione della stampa. Il BASIC AMSTRAD supporta totalmente lo standard industriale per la formattazione con il comando PRINT.

Nel caso in cui non venga specificato alcun formato, i dati sono stampati in 'formato libero', dove una virgola invia l'<elemento da stampare> all'inizio della successiva Zona di Stampa. Il punto e virgola serve per separare con un solo spazio gli <elementi da stampare>.

SPC (<espressione intera>) stampa il numero di spazi definito: nessuno se l'<espressione intera> è negativa. Se il numero di spazi da stampare è superiore al numero di caratteri disponibili sulla periferica (schermo, stampante, ecc.), questo viene ridotto al limite massimo consentito. Il comando SPC termina con l'invio automatico di un punto e virgola sul <canale>.

TAB (<espressione intera>) stampa degli spazi per muovere il cursore alla posizione specificata; se l'<espressione intera> è negativa la TAB stampa un solo spazio. Vale quanto detto per la SPC circa l'ampiezza della Zona di Stampa.

Se la posizione richiesta è inferiore a quella attuale, il cursore viene portato a capo e la tabulazione eseguita sulla nuova riga. Il comando TAB termina con l'invio automatico di un punto e virgola sul <canale>.

PRINT USING "<identificatore di formato>"

CAMPI NUMERICI

iden.	cifre	lungh.	definizione	esempio
#	1	1	campo numerico	####
.	0	1	virgola decimale	#.#
+	0	1	stampa il segno [+ o –] all'inizio o alla fine del numero	+##.## ##.##+
–	0	1	stampa il segno – dopo i numeri negativi, altrimenti spazio	##.##–
**	2	2	sostituisce il segno * agli eventuali spazi all'inizio del numero	**#.#
$$	1	2	stampa il segno $ prima del numero	$$##.##
**$	2	3	combina gli effetti dei due identificatori precedenti	**$#.##
,	1	1	precede la virgola decimale e stampa il segno [,] alla sinistra di ogni gruppo di tre cifre a partire dalla virgola decimale	####,.#
↑↑↑↑	0	4	formato esponenziale. Risultato allineato a sinistra.	#.#↑↑↑↑

STRINGHE

!	Stampa solo il primo carattere	!
N <spazi>N	Stampa una stringa di lunghezza pari al numero di spazi battuti più 2	\ \
&	Stampa la stringa così com'è.	&

CAPITOLO 9

ULTERIORI INFORMAZIONI SULLA PROGRAMMAZIONE

Argomenti trattati in questo Capitolo:

- Posizione del testo
- Caratteri di controllo
- Sistema Operativo
- Interrupt

9.1 — Posizione del cursore e codici di controllo

In molti programmi applicativi il cursore può essere posizionato al di fuori della finestra corrente, ma ci sono alcune operazioni che, prima di essere eseguite, forzano il cursore ad una posizione 'legale':

- scrittura di un carattere;
- evidenziazione del quadratino luminoso rappresentante il cursore;
- interpretazione dei codici di controllo contraddistinti con un asterisco nella lista riportata nelle pagine seguenti.

Descriviamo ora la procedura seguita per riportare il cursore ad una posizione legale:

a. Se il cursore si trova alla destra del margine destro, esso viene spostato sulla colonna più a sinistra della linea immediatamente sotto

b. Se il cursore si trova alla sinistra del margine sinistro, esso viene spostato sulla colonna più a destra della linea immediatamente sopra

c. Se il cursore si trova sopra il margine superiore, la finestra viene fatta scendere di una linea e il cursore posizionato sulla prima linea della finestra
d. Se il cursore si trova sotto il margine inferiore, la finestra viene fatta salire di una linea e il cursore posizionato sull'ultima linea della finestra.

Tenete presente che i controlli e le operazioni conseguenti seguono l'ordine usato nella descrizione precedente. Posizioni del cursore non legali possono essere lo zero o un numero negativo che indicano, rispettivamente, un fuori finestra sulla sinistra o rispetto al margine superiore.

Quando sullo schermo vengono inviati caratteri di codice compreso tra 0 e 31 (vedi Appendice III) non si ottiene alcuna visualizzazione, in quanto questi caratteri vengono interpretati come CODICI di CONTROLLO e possono anche causare il blocco del computer, se non usati con cognizione di causa. Alcuni di questi codici modificano il significato di uno o più dei caratteri che li seguono, che chiameremo parametri del codice.

I caratteri di controllo, quando inviati sulla pagina grafica, vengono stampati, a differenza di quanto accade in modo testo (vedi la descrizione del comando TAG nel Capitolo 8). Caratteri del tipo (&07 'BEL') provocheranno la visualizzazione di un simbolo convenzionale legato alla funzione svolta quando richiamati da tastiera (per il BEL, ad esempio, batterete [CTRL] G), mentre si limiteranno ad eseguire la funzione se inseriti in un comando del tipo PRINT CHR$(&07).

I codici evidenziati con il simbolo * nella tabella che segue forzano il cursore ad una posizione legale nella finestra corrente, pur potendo, a seguito dell'esecuzione dell'operazione richiesta, lasciare il cursore stesso in posizione non legale. Il valore dei codici è espresso prima in esadecimale (&XX) e poi in decimale.

CODICI DI CONTROLLO
(generalmente non sono accessibili da tastiera tramite il tasto [CTRL])

Valore		Nome	Parametro	Significato
&00	0	NUL		Nessun effetto. Ignorato
&01	1	SOH	0...255	Stampa il simbolo richiamato dal valore del parametro, permettendo la rappresentazione dei caratteri con codice compreso tra 0 e 31
&02	2	STX		Disabilita la visualizzazione del cursore
&03	3	ETX		Abilita la visualizzazione del cursore, di norma abilitato solo durante i periodi di attesa dell'input da tastiera

&04	4	EOT	0...2	Equivale a un comando MODE. Il valore del parametro viene calcolato in modulo 4
&05	5	ENQ	0...255	Invia al cursore grafico il carattere rappresentato dal parametro
&06	6	ACK		Abilita lo schermo (vedi &15, NAK)
&07	7	BEL		Emette un beep provocando lo svuotamento delle code sonore
&08	8 *	BS		Sposta il cursore indietro di un carattere
&09	9 *	TAB		Sposta il cursore in avanti di un carattere
&0A	10*	LF		Sposta il cursore verso il basso di una riga
&0B	11*	VT		Sposta il cursore verso l'alto di una riga
&0C	12	FF		Equivale ad un CLS
&0D	13*	CR		Sposta il cursore al margine sinistro della finestra sulla riga corrente
&0E	14	SO	0...15	Equivale al comando PAPER. Il parametro viene calcolato in modulo 16
&0F	15	SI	0...15	Equivale al comando PEN. Il parametro viene calcolato in modulo 16
&10	16*	DLE		Cancella il carattere corrente, portando la matrice all'Ink del Paper
&11	17*	DC1		Cancella la finestra a partire dal margine sinistro e fino alla posizione attuale del cursore, portando le matrici dei caratteri interessati all'Ink del Paper
&12	18*	DC2		Cancella la finestra a partire dalla posizione attuale del cursore e fino al margine destro, portando la matrice dei caratteri interessati all'Ink del Paper
&13	19*	DC3		Cancella la finestra a partire dall'inizio e fino alla posizione attuale del cursore, portando la matrice dei caratteri interessati all'Ink del Paper

&14	20*	DC4		Cancella la finestra a partire dalla posizione attuale del cursore e fino alla fine, portando la matrice dei caratteri interessati all'Ink del Paper
&15	21	NAK		Disabilita lo schermo, rendendo inoperativi tutti i comandi fino all'invio di un ACK
&16	22	SYN	0...1	Abilita (1) o disabilita (0) l'opzione di trasparenza. Il parametro è calcolato in modulo 2
&17	23	ETB	0...3	Il parametro è calcolato modulo 4: 0 seleziona gli Ink di default in grafica 1 esegue la XOR 2 esegue la AND 3 esegue la OR
&18	24	CAN		Scambia tra loro gli Ink del Paper e del Pen
&19	25	EM	0...255 0...255 0...255 0...255 0...255 0...255 0...255 0...255 0...255	Equivale al comando SYMBOL e definisce la matrice di un carattere. Richiede 9 parametri, di cui il primo indica il codice da definire e gli altri 8 descrivono la matrice: il bit più significativo del primo byte corrisponde al pixel dell'angolo in alto a sinistra della matrice, mentre il bit meno significativo dell'ultimo byte corrisponde al pixel dell'angolo in basso a destra
&1A	26	SUB	1...80 1...80 1...25 1...25	Equivale al comando WINDOW e definisce una finestra. I primi due parametri definiscono i margini sinistro (valore inferiore) e destro (valore superiore). Gli altri due parametri definiscono i margini superiore (valore minore) e inferiore (valore maggiore)
ESC				Nessun effetto. Ignorato
&1C	28	FS	0...15 0...31 0...31	Specifica una coppia di colori per lo Ink (vedi comando INK). Il primo parametro indica l'Ink (modulo 16) e gli altri due (modulo 32) definiscono i colori
&1D	29	GS	0...31 0...31	Come il comando BORDER, specifica una coppia di colori per il bordo. I due parametri sono calcolati modulo 32 ed indicano i colori

&1E	30	RS		Sposta il cursore sull'angolo in alto a sinistra della finestra
&1F	31	US	1...80 1...25	Come il comando LOCATE, posiziona il cursore alle coordinate date nella finestra. Il primo parametro indica la colonna e il secondo la riga

9.2 — Sistema Operativo

Il Sistema Operativo del CPC 464 opera in tempo reale e dirige il 'traffico' che avviene nel computer dalla partenza (input) all'arrivo (output).

In primo luogo il Sistema Operativo interfaccia l'hardware con l'interprete BASIC. Come esempio, nella funzione di flashing degli Ink l'interprete BASIC passa i parametri e il Sistema Operativo compie il lavoro, da una parte riconoscendo le richieste e dall'altra determinando la temporizzazione di questi eventi.

Il Sistema Operativo viene normalmente identificato con il termine 'firmware' e comprende le routine in linguaggio macchina richiamate dai comandi del BASIC.

Pur essendo praticamente impossibile bloccare il CPC 464 lavorando in BASIC, tranne che tramite il comando ON BREAK GOSUB, è relativamente facile provocare situazioni risolvibili solo con un reset (che, lo ricordiamo, cancella la memoria) quando si manipola il Sistema Operativo.

Se siete tentati di entrare nella memoria con delle POKE o di richiamare sottoprogrammi in linguaggio macchina con delle CALL, vi consigliamo di salvare tutto il salvabile prima di farlo. La descrizione dettagliata del Sistema Operativo esula dallo scopo di questo manuale ed è contenuta nella Guida al Firmware del CPC 464.

9.3 — Interrupt

Il CPC 464 utilizza ampiamente la gestione degli interrupt dello Z80 fino a mettere a disposizione dell'utente molte prestazioni multitasking, come quelle descritte a proposito della AFTER e della EVERY nel Capitolo 8. La priorità tra i temporizzatori è la seguente:

Break [ESC] [ESC]
Timer 3
Timer 2 (e le tre code sui canali sonori)
Timer 1
Timer 0

Tenete presente che lavorare con gli interrupt non è cosa semplice e immediata in quanto occorre sempre considerare le possibili conseguenze derivanti dallo stato di tutte le condizioni variabili al momento dell'interruzione. Le stesse subroutine di servizio delle interruzioni devono evitare interazioni non desiderate con lo stato delle variabili nel programma principale.

I canali sonori hanno interrupt indipendenti di uguale priorità. Quando si genera un interrupt di suono, questo non può essere a sua volta interrotto, permettendo ai comandi sonori di avere variabili in comune senza risentire degli effetti negativi più sopra menzionati. Gli interrupt di suono, una volta abilitati, vengono generati non appena la coda sonora presenta un posto libero, oppure quando inizia il suono successivo e in coda c'è posto per altri suoni. Una volta soddisfatta la richiesta, l'interrupt viene disabilitato, questo deve essere perciò riabilitato nel caso lo si voglia in seguito intercettare ancora. La generazione di un suono o il controllo dello stato di una coda disabilita gli interrupt.

Il fatto che la sequenza [ESC] [ESC] abbia priorità rispetto a tutti gli altri interrupt assicura la possibilità di interrompere l'esecuzione del programma BASIC senza perderlo, a patto che non siano stati attivati dispositivi ausiliari di protezione del programma.

9.4 — Linguaggio ASSEMBLER

La programmazione in linguaggio macchina richiede l'uso di un Assemblatore. Quello del CPC 464, acquisibile separatamente, comprende un assembler Z80 rilocabile con editor, disassembler e monitor.

CAPITOLO 10
I DISPOSITIVI DI INTERRUPT

Argomenti trattati in questo Capitolo:

- AFTER
- EVERY
- REMAIN
- Il clock di sistema

Dovrebbe essere ormai evidente come una delle principali innovazioni del software del CPC 464 consista nella possibilità di gestire gli interrupt da BASIC, che equivale a dire che il BASIC AMSTRAD è in grado di eseguire 'simultaneamente' un certo numero di operazioni distinte. Questa possibilità, spesso definita con il termine multitasking, può essere sfruttata dall'utente tramite l'uso dei comandi del BASIC AFTER e EVERY.

Anche la gestione delle code sonore e degli appuntamenti fra canali fornisce un chiaro esempio di multitasking.

Quando si parla di temporizzazione ci si riferisce sempre al clock di sistema, che consiste in un dispositivo al quarzo capace di sincronizzare tutti gli eventi che accadono nel corso delle elaborazioni. Tutte le funzioni hardware connesse al tempo sono controllate dal clock di sistema e rese facilmente accessibili tramite i comandi software AFTER e EVERY. Come spesso accade, il significato letterale di questi due termini in inglese indica chiaramente il tipo di funzione: DOPO (after) un intervallo specificato, oppure OGNI (every) n secondi il programma salta alla subroutine indicata e prosegue l'esecuzione.

10.1 — AFTER

Il CPC 464 mantiene un orologio in tempo reale. Il comando AFTER consente ad un programma BASIC di richiamare una subroutine dopo un intervallo di tem-

po determinato: dal momento che esistono quattro timer, è possibile associare una subroutine a ciascuno di essi.

AFTER <espressione intera>[, <espressione intera>] GOSUB <numero riga>

La prima <espressione intera> definisce, in cinquantesimi di secondo, il tempo che deve trascorrere prima di richiamare la subroutine, mentre la seconda identifica il timer da usare tramite un numero compreso fra 0 e 3 (0 per default). Allo scadere dell'intervallo di tempo definito, la subroutine viene automaticamente richiamata, proprio come se il programma incontrasse in quel momento un comando GOSUB. Alla chiusura della subroutine con il RETURN il programma principale riprende da dove era stato interrotto.
Ricordate che i timer hanno priorità decrescente dal numero 3 al numero 0.

Il comando AFTER può trovarsi in qualsiasi punto del programma e, una volta eseguito, azzera il timer ad esso associato. Dal momento che i timer possono essere usati anche dalla EVERY, ogni AFTER si sovrappone ad ogni precedente EVERY per lo stesso timer e viceversa:

```
10 MODE 1:X=0
20 AFTER 45 GOSUB 100
30 AFTER 100,1 GOSUB 200
40 PRINT "MICROSTAR"
50 WHILE X<100
60 LOCATE#1,30,1:PRINT #1,X:X=X+1
70 WEND
80 END
100 PRINT "computer"
110 RETURN
200 PRINT "e software"
210 RETURN
```

Notate l'uso delle due finestre che permette al programma principale (righe 50-80) di stampare utilizzando un posizionamento del cursore indipendente da quello delle subroutine di interrupt.

10.2 — EVERY

Con il comando EVERY il programma BASIC può richiamare una subroutine ad intervalli regolari. Vale quanto già detto in precedenza per i 4 timer:

EVERY (<espressione intera>[, <espressione intera>] GOSUB <n. riga>

La prima <espressione intera> definisce, in cinquantesimi di secondo, l'intervallo di tempo tra due successivi richiami della subroutine, mentre la seconda ha

lo stesso significato che nel comando AFTER. Anche in questo caso, una volta trascorso il tempo stabilito, il programma richiama automaticamente una subroutine come se incontrasse un comando GOSUB. Alla chiusura della subroutine con il RETURN, il programma principale riprende da dove era stato interrotto.

Ricordiamo quanto già detto a proposito delle priorità fra i timer: il 3 ha priorità maggiore e lo 0 quella minore. Allo scadere del tempo, il timer è azzerato e riprende il conteggio alla rovescia per il successivo richiamo.

Anche il comando EVERY si può trovare in qualsiasi punto del programma e si comporta nei riguardi dei timer come già detto dal paragrafo precedente:

```
10 MODE 1:X=0
20 P100=0:EVERY 10 GOSUB 100
30 P200=0:EVERY 12,1 GOSUB 200
40 PRINT "MICROSTAR"
50 WHILE X<200
60 LOCATE #1,30,1:PRINT #1,X:X=X+1
70 WEND
80 LOCATE 1,20:END
100 DI: PEN P100:LOCATE 1,2:PRINT "computer":EI
105 IF P100=0 THEN P100=1 ELSE P100=0
110 RETURN
200 PEN P200:LOCATE 1,3:PRINT "e software"
205 IF P200=2 THEN P200=3 ELSE P200=2
210 RETURN
```

Fate molta attenzione ai comandi di disabilitazione e abilitazione degli interrupt, DI e EI: tutte le istruzioni che seguono una DI non vengono interrotte da eventuali interrupt fino ad una EI. L'effetto pratico consiste nel ritardare l'interrupt del timer 1, che avrebbe priorità maggiore rispetto a quello del timer 0, fino a quando non termina la routine di servizio (righe 100 — 110) dell'interrupt del timer 0. Detta routine deve infatti aver il tempo necessario per invertire i codici della PEN e modificare la posizione del cursore prima che venga eseguito il comando PRINT.

10.3 — REMAIN

Questa funzione restituisce il valore del tempo rimanente su un determinato timer. Notate che disabilita il timer, restituendo 0 nel caso che questo risulti già disabilitato:

REMAIN (<espressione intera>)

Appendice 1

Coloro che si accostano per la prima volta ai computer di norma partono con lo stabilire alcuni punti di riferimento utili per ricavare una prima impressione sulle capacità e i limiti della macchina. Questa Appendice vuole essere una guida generale al 'che cosa' e al 'perché'.

Oliate le rotelle!!

Anche se l'unica ragione che vi ha spinto a dotarvi di un CPC 464 è stato il vostro desiderio di sfidare il computer nei più sofisticati videogiochi, non mancherete certo di chiedervi che cosa in effetti si intenda per tutto quanto va sotto il nome di hardware.

L'hardware, letteralmente 'parte dura', è tutto ciò che potete toccare e trasportare: la tastiera, il monitor, i cavi di connessione ecc.. In pratica, anche se molto banalmente, si può chiamare hardware quello che non è in maniera specifica software, cioè programmi, manuali e cassette.

Alcune caratteristiche del computer sono proprie dell'hardware. Ad esempio, mentre la capacità di produrre colori sul TV domestico o sul monitor è connessa alle caratteristiche hardware del computer, sarà poi compito del software sfruttare questa capacità per produrre caratteri e disegni colorati sullo schermo.

In altre parole, l'hardware si occupa di dirigere il fascio di elettroni sulla superficie elettroluminescente dello schermo, in modo da 'illuminarla', mentre il software aggiunge ordine e intelligenza comunicando all'hardware quando e come intervenire. Il software fornisce anche gli strumenti di temporizzazione e controllo per ottenere, ad esempio, l'effetto di un'astronave che decolla, oppure, molto più semplicemente, per far apparire un carattere sullo schermo alla pressione di un tasto.

D. Che cosa rende quindi un computer migliore di un altro?

L'hardware senza il software è privo di valore. È vero anche l'inverso, tanto che solo l'unione di questi due elementi attribuisce un valore al computer. Si possono fare alcune considerazioni di base che permettono di definire una scala di prestazioni dell'hardware e del software. Parlando di personal computer, i parametri generalmente accettati sono i seguenti:

1. La risoluzione, che identifica il più piccolo elemento rappresentabile sullo schermo

Si tratta di una combinazione di diversi fattori, tra cui il numero di colori utilizzabili, il numero di punti (pixel) singolarmente indirizzabili sullo schermo, il numero di caratteri rappresentabili su ogni linea dello schermo. Lasciamo a voi l'onere del confronto tra il CPC 464 e altri computer.

2. L'interprete BASIC

Praticamente tutti gli home computer comprendono un interprete BASIC, che permette all'utente di creare programmi sfruttando le capacità dell'hardware. Lo stesso linguaggio BASIC che vi viene fornito insieme alla macchina è un programma, estremamente complicato e sofisticato, sviluppato sulla base di oltre un milione di anni-uomo di esperienze maturate da quando il BASIC fu 'inventato' negli Stati Uniti. Il BASIC (Beginners All-purpose Symbolic Instruction Code) è sicuramente il linguaggio di programmazione più diffuso del mondo e, come tutti i linguaggi, comporta una varietà di dialetti.

Il dialetto del CPC 464 è uno tra i più diffusi e presenta caratteristiche di velocità veramente apprezzabili. Anche se, presumibilmente, per voi non ha molto interesse il fatto che un computer impieghi 1/20 di secondo a moltiplicare 3 per 5, mentre un altro può compiere la stessa operazione in 75/1000 di secondo, possiamo garantirvi che per un programma in cui si debbano eseguire alcune migliaia di moltiplicazioni la differenza fra 1/20 e 75/1000 di secondo assume importanza rilevante ai fini delle prestazioni.
Vi imbatterete certo di frequente nel termine Linguaggio Macchina, che sta ad indicare la 'lingua madre' della CPU, l'unica tramite la quale è possibile il colloquio. Il linguaggio macchina ha una velocità molto superiore a quella dell'interprete BASIC, da 5 a 15 volte, ma questo si paga caro in termini di tempo necessario per la scrittura dei programmi. Infatti, a parità di ogni altra condizione, sviluppare un programma in linguaggio macchina può richiedere un tempo da 5 a 50 volte superiore rispetto al BASIC.

Il BASIC AMSTRAD è tra i più veloci e completi, consentendo ai programmatori più esperti di evitare la maggior parte degli sprechi di tempo tipici dei linguaggi cosiddetti di alto livello e ottenere effetti grafici e musicali sorprendenti.

3. Espandibilità

Quasi tutti i computer si fanno notare per la possibilità di aggiungere alla configurazione di partenza alcune periferiche, come stampanti, joystick e dischi. Paradossalmente, alcuni degli home computer di maggior successo richiedono la presenza di elementi di espansione, noti come 'interfacce addizionali', anche per poter collegare un semplice joystick o una stampante.

Non sempre chi acquista un computer si pone il problema delle esigenze future, pur risultando evidente che, se il prezzo pagato comprende già tutte o alcune delle interfacce citate, esso può essere solo apparentemente superiore ad un prezzo che si riferisca a configurazioni base. Il CPC 464 include una porta paral-

lela Centronics per la stampante, una porta per la connessione dei Joystick e un bus di espansione utilizzabile per il collegamento del controller dischi, di ROM addizionali o di interfacce seriali, ecc..

Si definisce ROM (Read Only Memory = memoria a sola lettura) un dispositivo di memoria su chip usato per contenere programmi che una volta memorizzati non possono essere più cancellati. Lo stesso interprete BASIC fornito insieme con il computer è memorizzato in ROM. Altri programmi in ROM possono aggiungersi o sostituirsi alle ROM esistenti.

Le console dei videogiochi fanno uso di software su dispositivi chiamati cartridge, che altro non sono se non ROM alloggiate in simpatici contenitori di plastica dotati di uno speciale connettore che rende semplicissime le operazioni di collegamento e scollegamento. In questo modo, la ROM ha la stessa funzione della cassetta, ma la velocità con cui i programmi vengono caricati in memoria centrale è incomparabilmente superiore rispetto a quella del registratore che è nell'ordine dei minuti. Come ultima notazione, ricordate che non è possibile utilizzare una ROM per memorizzare informazioni provenienti dal computer.

Per concludere, le possibilità di espansione giocano un ruolo fondamentale nella valutazione di un computer e questo è stato tenuto ben presente nella progettazione del CPC 464.

4. Suono

Le prestazioni sonore di un computer determinano la possibilità di riprodurre in maniera accettabile gli effetti ottenibili con uno strumento musicale elettronico.

Il CPC 464 usa un generatore di suono a tre canali su 7 ottave, in grado di produrre effetti musicali di ottimo livello, con pieno controllo sugli inviluppi di tono e di volume. In aggiunta, il suono può essere riprodotto in stereofonia usando opportunamente i tre canali.

Gli effetti sonori trovano impiego immediato, ad esempio, per sottolineare le fasi salienti e i momenti più delicati in ogni azione di un qualsiasi gioco.

Lasciamo a voi decidere quale importanza attribuire a ciascuna delle caratteristiche fin qui descritte. Da parte nostra, ci auguriamo solo che vogliate provarle tutte, fino ad ottenere il massimo dal vostro computer.

Perchè no?

Viste tutte le possibilità della moderna tecnologia, gli utenti spesso si chiedono perché anche i computer più avanzati, come il CPC 464, sono apparentemente incapaci di riprodurre immagini del tipo di quelle ottenibili con qualsiasi apparecchio televisivo.

D. Perché, ad esempio, un computer non può produrre figure che si muovano sullo schermo in maniera naturale? Perché i movimenti vengono rappresentati con figure fatte da qualcosa di simile ai fiammiferi?

La risposta può essere semplice e complessa allo stesso tempo. In primo luogo, non dovete cadere nell'errore di ritenere che lo schermo del vostro computer abbia una definizione paragonabile a quella di un televisore. Un televisore opera con informazioni lineari in grado di descrivere una gamma virtualmente infinita di risoluzioni tra gli estremi luce/buio passando attraverso tutti i colori dello spettro luminoso. Questo significa che la 'memoria' necessaria a contenere un intero schermo TV dovrebbe avere una capacità di qualcosa come 20 volte superiore a quella richiesta da un normale display di computer.

Questa è solo una parte del problema, in quanto per animare delle figure è necessario che una così ampia memoria video venga manipolata ad elevata velocità. Si tratta di un'operazione possibile, ma solo per elaboratori che costano qualche migliaio di volte più di un home computer, per lo meno allo stato attuale delle cose.

Fino a quando il costo delle memorie ad alta velocità rimarrà elevato, i computer di piccole dimensioni dovranno accontentarsi di una memoria video relativamente limitata; questo significa bassa risoluzione e movimenti non del tutto armoniosi. Solo l'accurata progettazione dell'hardware e l'utilizzo di sofisticate tecniche di programmazione permettono di ottenere i migliori risultati in questa situazione. Siamo comunque ancora lontani dal vedere computer di basso costo produrre animazioni degne del meno animato fra i cartoni animati.

D. Perché non è possibile affrontare il computer semplicemente, battendo solo del testo?

Non lasciatevi ingannare dal fatto che il computer rassomiglia ad una macchina per scrivere dotata di un visore elettronico. Lo schermo non è un foglio di carta elettronica, ma una console di comando: cioè un oggetto che permette di comunicare degli ordini al computer.

Normalmente, il computer tenta di interpretare tutti i caratteri che vengono inseriti da tastiera come se si trattasse di istruzioni del BASIC. Alla pressione del tasto [ENTER] il computer controlla che tutto quanto è stato inserito abbia un senso in BASIC, altrimenti l'input verrà rifiutato e comparirà il messaggio:

    Syntax error

Esistono programmi di Elaborazione Testi appositamente concepiti per permettere al computer di trasformarsi in un sofisticato strumento di scrittura. Utilizzandoli, potrete sbizzarrirvi a scrivere tutto quanto vi verrà in mente senza preoccuparvi di altro, come se lo schermo fosse un foglio di carta e la tastiera quella di una macchina per scrivere.

Un computer 'sembra' composto di vari elementi che siamo abituati ad avere intorno in casa e nell'ambiente di lavoro, come il monitor simile ad un TV domestico, la tastiera e il registratore. Questa somiglianza riguarda solo l'esteriorità, in quanto all'aspetto familiare non corrispondono gli impieghi abituali.

Glossario

Alcuni dei termini correnti nel linguaggio dei computers spiegati all'utente del CPC 464.

Accoppiatore acustico

Chiamato anche Modem Acustico. Si tratta di un congegno elettronico che abilita il computer a comunicare sulla normale rete telefonica tramite collegamento con un qualsiasi telefono standard. In questo modo è possibile, ad esempio, comunicare sulla rete pubblica Videotel oppure scambiare software, informazioni e dati con altri utenti.

Accumulatore

Locazione di memoria all'interno del microprocessore, nel cuore, cioè, dell'intero sistema, usata per memorizzare temporaneamente un dato durante l'elaborazione. Interessa unicamente la programmazione in Linguaggio Macchina, mentre l'utente BASIC può tranquillamente ignorarne l'esistenza.

Alfanumerico

Aggettivo usato per definire in maniera generica un carattere o un insieme di caratteri, quando non si vuole o non si può specificare uno degli attributi 'alfabetico', 'numerico' o 'grafico'.

Algoritmo

Sequenza di operazioni logiche e/o aritmetiche atta a risolvere un problema definito.

ALU

Acronimo di Arithmetic and Logic Unit (Unità Aritmetica e Logica). Individua quella parte del microprocessore che si occupa, per l'appunto, delle operazioni aritmetiche e logiche. Interessa solo la programmazione in Linguaggio Macchina.

AMSOFT

Divisione della AMSTRAD specializzata nello sviluppo e fornitura di software, periferiche e pubblicazioni tecniche relative al CPC 464.

Animazione

Il far muovere grafici sullo schermo per simulare azioni dal vivo.

Applicativo (Programma –)

Programma che risolve i problemi specifici del signor Rossi; in contrapposizione a strumenti software di interesse e uso generalizzato, come un assembler o il driver della stampante.

Arcade (gioco)

Gioco di puro divertimento e di prontezza di riflessi, sullo stile di quelli delle Sale Giochi. Normalmente occorre difendersi da invasioni di alieni o evitare, e all'occorrenza uccidere, insaziabili mostri senza perdere le proprie 'vite'. Effetti spettacolari di grafica e suono, ma scarso valore educativo.

Architettura

Sistema che regola le relazioni tra il bus dati, le periferiche e la CPU. Ai lettori di questo glossario dovrebbe interessare veramente poco.

Argomento

Variabile indipendente. Ad esempio, nell'espressione X+ Y = Z, X e Y rappresentano gli argomenti.

Array

Letteralmente 'schiera', cioè una matrice a due dimensioni in cui gli elementi sono individuati dalle loro coordinate di riga e colonna.

Artificiale (Intelligenza –)

Tecnica di programmazione, per cui il programma impara sulla base di precedenti esperienze.

ASCII

Acronimo per American Standard Code for Information Interchange. Lo standard più diffuso per la rappresentazione di lettere, numeri e altri simboli così come forniti al computer da tastiera o richiamati dai comandi BASIC. L'elenco dei codici del CPC 464 è riportato nell'Appendice III.

Assembler

Linguaggio simbolico che facilita la programmazione in Linguaggio Macchina, in quanto le istruzioni sono rappresentate da codici mnemonici, che richiamano la funzione eseguita dalla routine corrispondente.

Avventura (Gioco di —)

Per alcuni è una religione, per altri una noia mortale! Si tratta di un gioco, normalmente di solo testo, in cui il giocatore deve trovare la via di uscita da un labirinto, combinando in maniera ... logica una serie di eventi apparentemente casuali.

A/D (Analogico/Digitale)

Identifica normalmente un processo di conversione da dati di tipo analogico, provenienti dal mondo esterno, a dati di tipo digitale, sui quali il computer può operare. Analogico si definisce un processo in cui le variazioni tra lo stato iniziale e quello finale avvengono in maniera graduale e non per commutazioni istantanee.

Base

Il fondamento di qualsiasi sistema di numerazione. I sistemi più utilizzati sono quello binario [Base 2], quello ottale [Base 8], quello decimale [Base 10] e quello esadecimale [Base 16]. All'Appendice II si trova una descrizione completa dei sistemi di numerazione.

BASIC

Significa: Beginner's All-purpose Symbolic Instruction Code. È un linguaggio di programmazione utilizzato dalla maggioranza degli home computer. Il BASIC è stato appositamente strutturato per risultare di facile apprendimento e utilizzo, in quanto permette di controllare il funzionamento di un programma a qualunque stadio del suo sviluppo.

Baud

L'unità di misura [bit per secondo] della velocità di trasferimento dei dati nei sistemi di comunicazione seriali.

BCD

Un sistema di codifica per i numeri decimali in cui ogni cifra è rappresentata da un gruppo di 4 bit [nibble].

Benchmark

Un programma standard che può essere eseguito con differenti computer per misurarne e confrontarne velocità, efficienza e precisione. Ad esempio, la radice quadrata di 99.999 elevato al quadrato.

Bit

L'abbreviazione di Binary digit, cioè cifra binaria. Può essere una qualsiasi delle due cifre ammesse nel sistema di numerazione binaria, 0 e 1.

Bit significant

L'informazione contenuta in un dato ha senso solo se viene considerato lo stato dei singoli bit che lo compongono e non il suo valore decimale.

Bit meno significativo

In un numero binario (vedi Appendice II), il bit meno significativo, a volte indicato con LSB, è quello che si trova all'estrema destra.

Boole - Algebra di

L'insieme delle regole logiche dove si possono avere solo due risposte: VERO o FALSO, solitamente rappresentate con 0 e 1.

Boot / Bootstrap

L'operazione di caricamento di programmi e Sistemi Operativi, eseguita solitamente da un programma residente in una ROM.

Buffer

Una zona di memoria utilizzata per la conservazione temporanea dei dati durante il loro trasferimento da un dispositivo all'altro. Ad esempio, tra la cassetta e

la memoria principale del computer. Un buffer permette lo scambio di informazioni tra dispositivi che operano a velocità differenti, quali, ad esempio, un modem o una stampante.

Bug

Un errore, di varia entità, dovuto ad un malfunzionamento del programma.

Bus

Insieme di connessioni, sia interne al computer che rivolte verso il collegamento dello stesso con l'esterno, che contengono varie informazioni circa lo stato della CPU, della memoria RAM e di altre componenti dell'hardware. Il bus del CPC 464 si trova sul più largo dei due connettori presenti sul retro del computer.

Byte

Un gruppo di 8 bit, l'unità di base della memoria indirizzabile da una CPU ad 8 bit. Ulteriori informazioni si trovano nell'Appendice II.

CAD

Acronimo di Computer Aided Design. Indica l'utilizzo delle capacità di calcolo e di rappresentazione grafica del calcolatore per la progettazione industriale.

CAE

Acronimo per Computer Aided Education. Usato per indicare l'uso del computer come strumento didattico. Più o meno equivalenti sono i termini CAI (Computer Aided Instruction) e CAL (Computer Aided Learning).

Canale

Area della memoria utilizzata per il trasferimento dei dati verso i dispositivi di output come lo schermo, la stampante e la cassetta.

Carattere

Qualsiasi simbolo rappresentabile da un computer. Può essere una lettera, una cifra o un simbolo grafico. (Vedi Appendice II).

Carattere grafico

Un carattere appositamente disegnato per facilitare la creazione di immagini particolari sullo schermo. Il CPC 464 dispone di un Set di caratteri grafici completo la cui descrizione si trova nell'Appendice III.

Caratteri (Set di —)

L'insieme dei caratteri, lettere, cifre e simboli grafici, rappresentabili da un computer o da una stampante. Il fatto che un carattere appartenga al Set di un computer non implica che lo stesso sia rappresentabile dalla stampante.

Caratteri (Stringa di —)

Un gruppo di caratteri.

Cartridge

Una ROM (vedi) normalmente racchiusa in un contenitore di plastica dotato di un connettore particolare per il collegamento diretto al computer. Il software su cartridge si carica molto rapidamente e facilmente, ma costa sensibilmente più del software su cassetta.

Cassetta

Supporto di memorizzazione usato dai registratori.

Chip

Erroneamente viene definito chip un qualunque tipo di circuito elettronico integrato. In realtà si tratta di una sottile 'fetta' di silicio trattato sulla quale il circuito integrato viene costruito.

Circuito Integrato

Un insieme di componenti elettronici miniaturizzati e assemblati su di un unico pezzo di silicio. Vedi anche 'Chip'.

Clock

La scansione del tempo, prodotta da un circuito di temporizzazione interno al computer, utilizzata per la sincronizzazione di tutte le componenti del sistema.

Codice

Frequentemente utilizzato dai programmatori come abbreviazione del termine Codice Macchina, sinonimo di Linguaggio Macchina.

Codice Macchina

Il linguaggio che viene direttamente compreso dalla CPU in quanto ogni istruzione è rappresentata da una sequenza di cifre binarie.

Codice a Barre

Un esempio si trova sulle etichette dei prezzi utilizzate per molti prodotti nei supermercati. Si tratta di un codice stampato che può essere letto dal computer tramite l'utilizzo di appositi strumenti quali un lettore ottico o un laser di bassa potenza.

Comando

Un'istruzione del linguaggio BASIC utilizzabile sia in Modo diretto che in Modo programma.

Compilatore

Un programma che converte una procedura scritta in linguaggio di alto livello, come il BASIC, in Codice direttamente eseguibile dalla CPU. I programmi compilati risultano molto più veloci di quelli interpretati, ma richiedono tempi di sviluppo notevolmente maggiori.

Computer (Generazioni di —)

Lo sviluppo della tecnologia utilizzata nella progettazione e nella costruzione dei computer ha segnato alcuni punti di riferimento secondo i quali i computer vengono divisi in generazioni.

CP/M

Acronimo di Control Program for Microcomputer, è lo standard de facto fra i Sistemi Operativi su disco per computer con CPU ad 8 bit.

CPU

Acronimo per Central Processing Unit. È il cuore e il cervello di un computer: si occupa dell'esecuzione vera e propria dei comandi ricevuti dalle periferiche del sistema e ne gestisce interamente le risorse.

Cursore

Un indicatore che segnala la posizione alla quale verrà visualizzato il prossimo carattere stampato sullo schermo.

Cursore grafico

Simile al cursore testo, ma rivolto all'indirizzamento nella pagina grafica. Non si tratta di un carattere stampabile, ma è solo un'espressione concettuale utilizzata per indicare il punto da cui avrà inizio una successiva operazione grafica.

Cursore (tasti di controllo del —)

Tasti che permettono di spostare il cursore in ogni direzione, sono frequentemente usati nei giochi e si distinguono per le frecce stampate sulla parte alta del tasto.

Database

Insieme di dati, anche eterogenei, accessibili in più modi.

Dati (Raccolta di —)

Termine usato letteralmente per significare la raccolta di dati da qualsiasi sorgente esterna collegata in qualche modo al computer.

Debugging

Il processo di ricerca e correzione degli errori nel programma.

Default (Valore di —)

Letteralmente si potrebbe tradurre con 'Valore di difetto'. Indica qualsiasi valore assegnato automaticamente dal computer ad un variabile di qualsiasi tipo, in mancanza di un esplicito assegnamento. Ad esempio, dato che all'accensione viene selezionato il Modo di schermo numero 1, si dice che 'Il Modo 1 è quello di default'.

Diagnostica

Normalmente si comprende sotto questo termine l'insieme delle routine di software destinate a controllare, riconoscere e segnalare gli eventuali malfunzionamenti del sistema e/o dei programmi.

Diagramma di flusso

Rappresentazione schematica dei vari blocchi logici di cui è composto un programma che tiene conto dell'andamento dello stesso.

Digitale

Variazione delle quantità per passi discreti in contrapposizione a processi continui. L'opposto di analogico.

Digitizer

Normalmente si indica con questo termine un tipo di tavoletta grafica che permette di convertire dati analogici in forma digitale.

Disco

Superficie piana circolare di plastica magnetizzata su una o entrambe le facce, utilizzata per memorizzare informazioni. I dischi dei computer sono alloggiati in contenitori protettivi, che li preservano dalla polvere. Vedi anche Floppy disk e Winchester.

Disk drive

Letteralmente 'giradischi', cioè quella periferica che serve per leggere un disco o scrivere sopra di esso, facendolo ruotare ad alta velocità sotto le testine di lettura/scrittura.

Documentazione

L'insieme dei manuali che illustrano l'hardware e il software di un computer.

DOS

Acronimo di Disk Operating System. Si definisce in questo modo l'insieme delle routine che sovraintendono a tutte le operazioni dei disk drive.

Download

Il processo di trasferimento di informazioni da un computer ad un altro.

Editing

Processo tramite il quale si correggono o si variano delle righe su video, siano esse di programma o di testo.

Editor

Programma di sistema, normalmente residente in ROM, che si occupa della gestione dell'editing.

EPROM

Acronimo per Erasable Programmable Read Only Memory. Simile alla PROM (vedi), con la differenza che i dati contenuti nella memoria possono essere cancellati utilizzando raggi ultravioletti. Nel caso che il processo di cancellazione sia invece elettrico, allora si usa il termine EEPROM.

Espressione

Una formula più o meno complessa che permette nell'ambito del programma di eseguire calcoli su dati, la cui natura è generalmente definita dall'espressione.

File

Spesso si traduce con 'flusso'. Indica un insieme di informazioni generalmente memorizzate su disco o cassetta, anche se alcuni computer permettono di gestire file in memoria RAM.

Firmware

Il software contenuto in memorie ROM. Può essere visto come un incrocio fra l'hardware e il software propriamente detti.

Floppy disk

Un disco magnetico, protetto da un involucro plastico, che permette l'archiviazione di grossi volumi di dati e programmi ad alta velocità. I diametri standard per i floppy disk sono 5.25 e 8 pollici.

Forth

Linguaggio di programmazione veloce e compatto dotato di alcune caratteristiche proprie del Linguaggio Macchina, ma anche di altre tipiche dei linguaggi di alto livello.

Generatore sonoro

La parte di un computer in grado di produrre suoni e rumori.

Grafica

La possibilità di molti computer di tracciare sullo schermo segmenti, curve e altre forme geometriche. Disponendo di una stampante e del software necessario è possibile ottenere copie su carta di tutto quanto visualizzato sullo schermo.

Handshaking

Sequenza di segnali che permettono l'inizio e la sincronizzazione delle comunicazioni fra un computer e periferiche o tra due computer.

Hard Copy

Riproduzione su carta di quanto visualizzato sullo schermo, o meglio contenuto nella memoria video.

Hardware

La parte elettronica e meccanica di un sistema di elaborazione, praticamente tutto ciò che non è software nè firmware.

Indirizzo

È quel numero che identifica, nell'ambito di una istruzione, una particolare 'cella' di memoria, consentendo di selezionarla e di leggerne il contenuto. Nel caso di memoria RAM è anche possibile riscrivere il contenuto, dopo eventuale modifica, allo stesso indirizzo da cui era stato prelevato.

IEEE–488

Uno degli standard di interfacciamento per il collegamento dei computer con le periferiche. Simile per molti versi, ma non totalmente compatibile, con lo standard Centronics.

Informazione (tecnologia della –)

Tutto ciò che si riferisce all'utilizzo di apparecchiature elettroniche nell'elaborazione delle informazioni e nelle comunicazioni. Ad esempio: elaborazione testi, banche dati, Sistema VIDEOTEL ecc..

Inizializzazione

L'accensione del sistema, o la dichiarazione di valori specifici per variabili prima che inizi l'esecuzione del programma. Ad esempio, la dichiarazione di variabili Intere ecc..

Input

Tutto quanto viene inserito nella memoria del computer da tastiera, da cassetta, da floppy, o da altre fonti.

Interattivo

Generalmente con questo termine si indicano i programmi che 'comunicano' in tempo reale con l'utente chiedendogli di effettuare delle scelte o di inserire dati che influenzano l'esecuzione del programma stesso.

Interfaccia

Ciò che permette al computer di trasmettere e ricevere dati. Il CPC 464 dispone della tastiera per l'input, dello schermo per l'output, e del connettore posteriore per l'interfacciamento con varie periferiche.

Interfaccia parallela

Un'interfaccia è 'parallela' quando ogni linea di dato presente sul connettore del computer viene collegata con quella corrispondente della periferica. La trasmissione parallela dei dati è più veloce di quella seriale, in quanto in quest'ultima ogni dato viene 'preparato' prima dell'invio e vengono inseriti stati di attesa nella comunicazione per far si che computer e periferica siano totalmente sincronizzati. L'interfaccia stampante del CPC 464 è del tipo parallelo.

Interfaccia seriale

Nonostante questo termine venga quasi sempre usato riferendosi ad un interfaccia RS232C, esistono altri standard per le interfacce di comunicazione seriali.

Interfaccia uomo - macchina

Dispositivo che permette l'interazione tra l'operatore e il computer, come la tastiera, lo schermo ecc..

Interprete

Quella parte del software di sistema che converte i comandi di un linguaggio di alto livello, come il BASIC, in istruzioni eseguibili dalla CPU.

Introduzione

Una serie di regole e concetti per illustrare all'utente alcuni aspetti della programmazione.

Istruzione

Un comando che provoca l'esecuzione da parte del computer di una particolare azione. Una sequenza di istruzioni forma un programma.

Istruzioni (Set di —)

L'insieme delle istruzioni direttamente eseguibili dalla CPU. Ogni comando di un linguaggio simbolico deve essere trasformato in una sequenza di istruzioni riconoscibili dalla CPU. Per questo motivo, una sola istruzione di un linguaggio di alto livello può richiedere l'esecuzione di molte istruzioni del set della CPU.

I/O

Acronimo di Input/Output.

Iterazione

Ripetizione di più operazioni semplici sino al verificarsi di una data condizione.

Joystick

Accessorio che viene generalmente utilizzato nei giochi per rendere più semplice e intuitivo il controllo di oggetti in movimento sullo schermo.

K

Abbreviazione di 'kilo', cioè migliaia, frequentemente usata per la misurazione della capacità di memoria di un computer in termini di 'kilobyte'. In realtà 1 kilo corrisponde a 1024 unità e non a 1000.

Linguaggio di alto livello

Linguaggio che utilizza un codice di facile comprensione per l'uomo. Simili linguaggi, come il BASIC, risultano più lenti del Codice Macchina, ma molto semplici da imparare.

Linguaggio di basso livello

Linguaggio di programmazione, come l'Assembler, in cui ogni istruzione corrisponde ad una di quelle eseguibili dalla CPU.

LISP

Acronimo formato da LISt Processor Language. È un linguaggio di alto livello utilizzato negli studi sull'intelligenza artificiale.

Logica

Componenti elettronici che eseguono le operazioni logiche elementari che compongono ogni operazione eseguita dal computer.

LOGO

Linguaggio di alto livello particolarmente utilizzato per scopi educativi.

Linguaggio di programmazione

Insieme di comandi e di regole sintattiche che permettono la stesura di programmi.

Loop

Processo che viene ripetuto più volte sino a quando non viene soddisfatta una determinata condizione.

LSI

Acronimo per Large Scale Integration. Indica il livello di integrazione di un circuito integrato, cioè il rapporto tra le sue dimensioni e le funzioni che è in grado di svolgere.

Matrice

L'insieme dei punti che definiscono la forma di un carattere, sia esso visualizzato sullo schermo o stampato su carta. Questo termine è usato in matematica, così come in informatica, per indicare vettori a più dimensioni.

Matrice di carattere

La matrice dei punti utilizzata per rappresentare un carattere sullo schermo. (Vedi Appendice II).

Matrice di punti

Griglia rettangolare nella quale vengono rappresentati i caratteri tramite la selezione di alcuni punti.

Memoria

La zona in cui il computer conserva dati e programmi. Esistono memorie di diverso tipo, principalmente: RAM (Random Access Memory) dove ogni informazione può essere sia letta che scritta e ROM di cui è possibile solo la lettura. Dischi e nastri sono classificati sotto il nome di memorie di massa.

Memoria (mappa della —)

La descrizione delle zone in cui la memoria è divisa e dello scopo a cui ciascuna di queste è dedicata.

Menu

Lista delle funzioni eseguibili da un programma nella quale l'utente deve scegliere quella desiderata.

Microprocessore

Circuito integrato in grado di gestire la memoria, controllare i dispositivi di I/O ed eseguire programmi in Codice Macchina.

Modem

Modulatore-Demodulatore che permette l'allacciamento del computer alla linea telefonica o ad altre linee di trasmissione seriali. Vedi anche Accoppiatore acustico.

Modo grafico

Con i primi personal computer era necessario selezionare il modo grafico prima di eseguire comandi grafici. Oggi, i computer sono in grado di visualizzare simultaneamente sia caratteri che grafica.

Modulatore RF

Indica il modo in cui i segnali video provenienti dal computer vengono codificati e trasmessi all'ingresso di un apparecchio TV standard.

Monitor

Lo schermo di un terminale.

Monitor (programma —)

Permette di accedere ad alcune delle funzioni di base della CPU e facilita la correzione dei programmi scritti in Codice Macchina.

MSB

Acronimo per Most Significant Bit. Il bit più significativo di un numero binario è quello che si trova all'estrema sinistra.

Mouse

Un dispositivo che viene generalmente usato per permettere il controllo del cursore con il semplice movimento della mano.

Nibble

La metà di un byte, cioè 4 bit.

Numero casuale

Numero generato dal computer secondo particolari algoritmi che non permettono la ripetizione dello stesso numero e nemmeno la previsione del prossimo numero.

Numero di riga

Il BASIC ed altri linguaggi di programmazione fanno uso dei numeri di riga per stabilire la sequenza d'esecuzione del programma.

Numero in virgola fissa

Un numero in cui la virgola decimale conserva sempre la medesima posizione, sia durante la manipolazione che la memorizzazione.

Numero in virgola mobile

Un numero Reale in cui la virgola decimale viene posizionata conseguentemente alle operazioni svolte. L'opposto di un numero in virgola fissa.

Numero Binario

Un numero rappresentato in Base 2. Tali numeri vengono preceduti sul CPC 464 dal prefisso &X. Ad esempio, &X0101 = 5 [decimale].

Numero Intero

Un numero privo della parte frazionaria, dove tutto quanto si trova alla destra della virgola decimale viene scartato.

Numero privo di segno

Numero rappresentato sempre senza segno, non importa se positivo o negativo.

Numero Reale

Numero formato da una parte intera e da una frazionaria. Una variabile può essere considerata Reale anche se in alcuni punti del programma contiene un valore Intero.

OCR

Acronimo di Optical Character Recognition. Una tecnica che permette di leggere e riconoscere caratteri scritti o stampati e di convertirli in un formato comprensibile al computer.

Off line

Indica che una periferica, solitamente un terminale o una stampante, è connessa al computer in modo passivo, non è cioè in grado di comunicare direttamente con l'unità centrale di elaborazione.

On line

L'opposto di Off line.

Operatore

La parte di un'espressione aritmetica che esegue un'operazione su due numeri.

Ottale

Sistema di numerazione con Base 8, in cui vengono utilizzate le cifre dallo 0 al 7.

Output

Tutto ciò che viene inviato dal computer verso l'esterno.

Paddle

Un accessorio che ha la stessa funzione del joystick, ma invece della leva ha una manopola da ruotare.

Parola chiave

Una 'parola' il cui uso è riservato per una specifica funzione, generalmente un comando del linguaggio di programmazione.

Parola riservata

Parola che ha un significato particolare nel linguaggio di programmazione e non può quindi essere utilizzata per altri scopi. Ad esempio, il comando NEW del BASIC è una parola riservata e non può essere usata come nome di variabile.

PASCAL

Linguaggio 'compilato' di alto livello che facilita la programmazione strutturata e genera un codice molto veloce.

PEEK

La funzione del BASIC che restituisce il contenuto della locazione di memoria specificata.

Penna ottica

Dispositivo di input costituito da una penna la cui posizione sullo schermo o su di una tavoletta viene riconosciuta dal computer.

Periferica

Qualsiasi dispositivo possa essere connesso al computer per espanderne le possibilità.

Pixel

Il più piccolo elemento singolarmente indirizzabile sullo schermo.

Plotter

Periferica in grado di stampare grafici molto precisi utilizzando una penna guidata da bracci meccanici. Utilizzato per il disegno tecnico.

POKE

Il comando del BASIC che scrive nella locazione di memoria specificata un determinato valore.

Porta

Utilizzata per inviare o ricevere dati da un'interfaccia.

Portabilità

Oltre al significato letterale, indica la possibilità del software di essere eseguito da computer differenti fra loro. Generalmente la portabilità del software è garantita solo dall'uso di Sistemi Operativi standard, quali il CP/M della Digital Research.

Porta logica

Le porte logiche permettono il passaggio dei dati solo quando alcune condizioni vengono soddisfatte; ne esistono di vari tipi, i più diffusi sono: OR, AND e XOR. Vedi anche Algebra di Boole.

PROM

Acronimo per Programmable Read Only Memory. Una memoria che una volta programmata non può più essere cancellata. Vedi anche EPROM.

PROMPT

Simbolo o messaggio che viene visualizzato per indicare che il computer, o il programma in esecuzione, è pronto per ricevere l'inserimento di un comando o di un dato. Il BASIC utilizza un semplice punto interrogativo [?] quando è in attesa di un input, e il messaggio 'Ready' per indicare che è pronto a ricevere un nuovo comando.

Programma

L'insieme delle istruzioni che vengono date al computer sotto forma di comandi perché questo esegua un determinato lavoro. Con programma si indica una qualsiasi sequenza di istruzioni senza considerare il loro numero o la complessità delle funzioni che svolgono.

Programmazione strutturata

Tecnica di programmazione che permette la soluzione di un problema tramite la sua scomposizione in più sotto-problemi semplici.

PSU

Acronimo di Power Supply Unit (Alimentatore). Il dispositivo che converte la tensione di rete nel voltaggio richiesto dal computer e dalle periferiche ad esso connesse.

Quinta generazione

Si riferisce principalmente a grossi computer, ancora in fase progettuale, che dovrebbero possedere la facoltà di autoprogrammarsi sfruttando le tecniche dell'intelligenza artificiale.

RAM

Acronimo per Random Access Memory. Un tipo di memoria che può essere sia letta che scritta, ma perde il suo contenuto se non alimentata. Viene utilizzata per la conservazione dei programmi, e relativi dati, durante la loro esecuzione.

Raster

Sistema di gestione dello schermo che utilizza un metodo di scansione orizzontale del video.

Ricorsività

Sequenza di operazioni ripetute ciclicamente, a volte non è possibile determinare a priori quante, dove ognuna è richiamata e influenzata dalla precedente.

Refresh

Aggiornamento delle informazioni, sia sullo schermo che in memoria. Non deve distruggere i dati, ma semplicemente 'riscrivere' quanto era presente sullo schermo o in memoria.

Registro

Locazione di memoria interna alla CPU che viene da questa utilizzata per la memorizzazione temporanea dei dati.

REM

Comando del BASIC che non ha alcun effetto sull'esecuzione del programma, ma può essere inserito nel listato per indicare la funzione svolta da un blocco dello stesso.

Rete

Il collegamento tramite linee di trasmissione differenti fra due o più computer in modo da permettere lo scambio di informazioni fra questi.

Risoluzione

Indica il numero di punti singolarmente indirizzabili sullo schermo. Viene scorrettamente utilizzato per definire la capacità di un computer di eseguire calcoli con numeri particolarmente grandi.

Riconoscimento vocale

La conversione di parole in un formato comprensibile dal computer.

ROM

Acronimo per Read Only Memory. Memoria a semiconduttore che una volta scritta non può essere cancellata, nè riprogrammata.

Routine

Parte di programma che esegue un lavoro 'di routine'. Può trattarsi di un sottoprogramma incluso nel programma principale o di un modulo a sè stante che può essere utilizzato in più programmi. Ad esempio, un programma che trae dal clock di sistema un orologio può essere considerato una routine.

RPN

Acronimo di Reverse Polish Notation. Un metodo per la scrittura delle espressioni matematiche che permette l'eliminazione delle parentesi.

RS232C

Indica uno standard per le interfacce di comunicazione seriale. Simili interfacce devono essere 'configurate' a seconda della periferica connessa, al contrario di quanto avviene per le interfacce parallele Centronics dove le connessioni sono rigidamente legate allo standard.

Rumore

Il CPC 464 ha la possibilità di generare, oltre a suoni polifonici, anche del rumore in modo da ottenere effetti sonori come esplosioni ecc.

Screen Editor

Un programma che permette l'inserimento e la modifica di testo e programmi.

Scrolling

Indica il modo in cui avviene lo spostamento di tutto quanto rappresentato sullo schermo quando il cursore raggiunge l'ultima riga disponibile ed è necessario creare lo spazio per la stampa di altri caratteri.

Separatore

Carattere utilizzato per delimitare le parole riservate separandole dagli altri elementi del programma.

Simulazione

Tecnica che permette di emulare e analizzare con il computer processi reali, come avviene con i Simulatori di volo ecc..

Sintesi vocale

Generazione della voce umana tramite l'utilizzo di hardware e software.

Sistma Binario

Il sistema di numerazione che ha Base 2, in cui cioè tutti i numeri vengono rappresentati con sequenze di 0 e 1. [vedi anche Base].

Sistema decimale

Sistema di numerazione in Base 10, che utilizza le cifre da 0 a 9 per rappresentare le unità, le decine, le centinaia, le migliaia e così via.

Sistema Esadecimale

Sistema di numerazione con Base 16, in cui vengono utilizzate tutte le cifre dallo 0 al 9 e le lettere dalla A alla F. Nel CPC 464 viene indicata con il prefisso & o &H.

Sistema Operativo

L'insieme delle routine che gestiscono la memoria del computer e supervisionano l'esecuzione dei programmi utente.

Software (Progettazione del —)

Indica gli studi che vengono effettuati su di un problema prima di passare alla stesura del programma che ne permette la soluzione tramite l'uso del computer.

Sovrascrittura

La cancellazione di una zona di memoria sostituendo il suo contenuto con nuovi dati.

Stampante

Periferica in grado di stampare su carta testo ricevuto dal computer.

Stampante a Margherita

Stampante in grado di produrre documenti con qualità di stampa paragonabile a quella di una macchina per scrivere. In altre parole, i caratteri sono 'pieni' invece che disegnati per punti.

Spreadsheet

Programma che permette di inserire dati in una matrice e stabilire tra essi relazioni di tipo matematico. Il cambiamento di uno dei dati legati da una formula provoca l'aggiornamento di tutti i risultati.

Sprite

Carattere definito dall'utente il cui movimento sullo schermo è controllato da appositi elementi dell'hardware e del software permettendo di ottenere facilmente effetti di animazione.

Stack

Zona di memoria che la CPU riserva per la memorizzazione temporanea del contenuto dei registri. I dati posti nello stack possono essere riletti in ordine inverso rispetto a quello di scrittura.

Statement

Termine utilizzato per indicare una o più istruzioni di un programma.

Stringa

Gruppo di caratteri sia numerici che alfabetici; anche nel caso in cui una stringa sia composta solo di cifre queste verranno trattate come caratteri, senza considerarne cioè il valore numerico.

Subroutine

Vedi routine.

Syntax error

Messaggio visualizzato quando una istruzione del programma è stata utilizzata senza seguire le regole sintattiche del BASIC. Vedi l'Appendice VIII.

Tasti di funzione

Un tasto a cui è stata associata una funzione specifica, un comando del BASIC o altro. Il CPC 464 dispone di alcuni tasti definibili a cui è possibile associare stringhe lunghe sino a 32 caratteri e i cui utilizzi possono essere i più vari.

Tastiera

La matrice dei tasti alfanumerici che permette l'inserimento di dati nel computer.

Tastiera QWERTY

Termine usato per indicare che una tastiera segue la disposizione standard dei tasti.

Tastierino Numerico

La zona della tastiera in cui sono raccolti i tasti numerici, facilitando così l'inserimento di numeri.

Tavola della verità

Tabella in cui vengono stabiliti i risultati delle operazioni logiche secondo la logica del 'VERO' o 'FALSO', dove il 'VERO' ha valore 1 e il 'FALSO' ha valore 0.

Tavoletta grafica

Periferica che permette di disegnare su un piano qualunque figura, permettendo poi di inviare il risultato al computer per successive elaborazioni. È una forma di conversione Analogico/Digitale.

Tempo reale

Indica un modo particolare in cui il computer fornisce l'ouput: i risultati vengono mostrati appena pronti, senza attendere il termine del processo che li ha forniti.

Terminale

Periferica, costituita da uno schermo e da una tastiera, utilizzata per la comunicazione con il computer.

Terminale video

Periferica di computer. Il video può essere 'stupido' o 'intelligente', a seconda che possegga o meno capacità di elaborazione proprie sfruttabili anche quando non è stabilita la connessione fisica con il computer.

Troncamento

Indica la riduzione della lunghezza di una stringa tramite l'eliminazione di caratteri all'inizio o alla fine di questa. Se eseguito su numeri può comportare anche l'arrotondamento, oppure semplicemente lo scarto di alcune cifre.

UDK

Acronimo di User Definable Keys. Si tratta di tasti a cui è possibile associare stringhe per compiere svariate operazioni.

Utility

Programma che svolge una funzione d'uso frequente, come l'ordinamento di un insieme di dati, la copia di file su disco ecc..

Variabile

Elemento fondamentale di ogni programma; viene identificato da un nome e il suo valore può essere modificato durante l'esecuzione del programma.

Appendice II

Un'occhiata al mondo dei computer

Chi ha problemi di gergo?

Come tutti i settori specializzati, anche quello dei computer ha un gergo proprio che permette di esprimere concetti a volte complessi con poche parole. L'alto livello tecnologico non è il solo colpevole della creazione di questa particolare terminologia: la maggior parte di noi si scontra con le difficoltà di comprensione del gergo utilizzato in ognuna delle principali professioni e attività dell'uomo.

Non sono tanto le parole a creare confusione, ma piuttosto il modo in cui queste vengono usate. Quasi tutte le persone che acquisiscono una certa familiarità con la terminologia del mondo dei computer danno alle parole un significato chiaro e preciso in modo da ridurre le difficoltà di comunicazione.

Il linguaggio del mondo dei computer non è comunque uno stile letterario, ma una scienza esatta che, oltre ad una sintassi precisa, conserva una struttura delle comunicazioni immediata senza permettere ambiguità circa il significato dei termini usati. I docenti di informatica non hanno ancora dato una definizione esplicita di cosa si debba intendere per "semantica di un programma".

Detto ciò, nonostante il significato di un programma possa risultare più o meno ovvio, esistono molti aspetti che possono essere analizzati come l'eleganza o la 'pulizia' dello stile di programmazione. Ora che le prime incomprensioni provocate dalla rivoluzione dei micro computer sono state superate, viene attribuita grande importanza ad un approccio formale alla costruzione del programma.

I giovani imparano facilmente a utilizzare un computer, aiutati dalla precisione e semplicità dei concetti e dal modo in cui questi possono essere trasmessi. Difficilmente troverete un avvocato di 10 anni, ma è probabile che incontrerete molti programmatori giovanissimi.

Le basi del BASIC

Praticamente tutti gli home computer sono dotati del linguaggio di programmazione BASIC, il più semplice da imparare e quello che permette di scrivere programmi senza avere particolari basi teoriche. Nonostante questo molti programmi estremamente complessi e sofisticati sono stati scritti in BASIC.

Comunque, non c'è dubbio che la semplicità d'apprendimento e d'uso del BASIC abbiano contribuito a renderlo praticamente universale.

D'ora in poi utilizzeremo alcuni dei termini propri del mondo dei computer; nel caso che il loro significato non vi risultasse chiaro riferitevi al GLOSSARIO presente alla fine di questo manuale.

Il BASIC è un linguaggio composto da un insieme di comandi che vengono interpretati durante il corso del programma, eseguendo diversi tipi di operazioni sui dati. Mentre una lingua come l'italiano ha un vocabolario di 5—6 mila persone, senza considerare le varie accezioni che ognuna di queste può avere, il BASIC è formato da meno di 200 comandi. Un programma scritto in BASIC deve seguire regole sintattiche ben precise e ogni tentativo di comunicare con il computer in forma discorsiva provocherà la visualizzazione del messaggio:

Syntax error

Il BASIC è principalmente dedicato ad elaborazioni numeriche, cioè all'esecuzione di operazioni su numeri e la maggior parte dei comandi è sostanzialmente un'estensione dei normali operatori matematici. Il più importante concetto che ogni principiante deve assimilare è che un computer è in grado di operare solo su dati numerici, dato che tutte le informazioni inviate alla CPU vengono a questa fornite in tale forma.

Anche volendo memorizzare in un computer tutta l'opera del Manzoni ogni lettera verrà convertita in un codice numerico, in quanto solo se così codificata potrà essere elaborata come richiesto.

Il BASIC quindi considera anche le parole come numeri su cui è possibile compiere operazioni matematiche come addizioni e sottrazioni, e operazioni logiche per confrontare tra loro due dati e stabilire se sono uguali o diversi, quale è il maggiore e quale il minore.

Tramite il programma il computer scompone ogni elaborazione in una serie di operazioni che accettano due soli risultati: SI e NO. Ad esempio, una moltiplicazione viene trasformata in una serie di addizioni: per moltiplicare 35 × 10 il BASIC sommerà 10 volte 35 a se stesso. In un registro della CPU si memorizza il dato 10, in un altro il 35. Si somma 35 al valore precedente e si decrementa di un'unità il registro che inizialmente conteneva il 10, sino a quando tale registro non raggiunge il valore 0. A questo punto un'altra parte della CPU si occupa dell'output del risultato.

Questo procedimento può sembrare macchinoso, ma non per un computer che è principalmente un oggetto in grado di compiere ad elevatissima velocità e con estrema precisione operazioni ripetitive. Così il BASIC traduce i comandi del programma nel codice che può essere compreso ed eseguito dalla CPU. La logica di un computer prevede solo due possibilità: SI e NO, che vengono rappresentate in notazione binaria rispettivamente come 1 e 0. Nella logica di Boole esistono solo due condizioni: VERO e FALSO, non esiste niente di simile a 'forse' o 'può darsi'.

Nel passaggio da uno stato '0' ad uno stato '1', o da 'VERO' a 'FALSO', si trova l'essenza del termine 'digitale'. In natura la maggioranza dei cambiamenti avvie-

ne in più fasi, con passaggi graduali. In campo digitale il passaggio da uno stato all'altro avviene in tempo nullo. In realtà il tempo di commutazione non è mai pari a 0 a causa delle caratteristiche fisiche dei semiconduttori che comportano ritardi detti di 'propagazione'. È a causa del sommarsi di questi ritardi che i computer impiegano sempre un certo tempo nell'elaborazione dei dati prima di rispondere ad una domanda.

In ogni caso, il computer deve attendere la fine di un'elaborazione prima di iniziarne una nuova, quindi un certo ritardo nella risposta esisterà sempre.

I processi digitali ammettono due soli stati ben distinti, non è possibile avere una condizione diversa da una delle due concesse, o meglio tutto ciò che non è 1 nè 0 è totalmente privo di senso. Questo garantisce risultati determinati e univoci. Il fatto che i computer risultino imprecisi nelle operazioni matematiche è dovuto alle limitazioni nel formato dei numeri elaborabili, a volte è perciò necessario troncare i dati perchè possano essere memorizzati nello spazio disponibile causando così errori di arrotondamento. Ad esempio, 999.999.999 diventa 1.000.000.000.

Come è possibile contare oltre 1 in un mondo dove le due sole cifre disponibili sono 0 e 1?

Bit & Byte

Tutti noi siamo abituati a utilizzare i numeri del sistema decimale dove la base di riferimento è il numero 10; in effetti le cifre utilizzabili sono dieci, quelle dallo 0 al 9. Il sistema di numerazione binario è tale per cui le cifre utilizzabili sono 0 e 1 e l'unità del sistema è il bit, abbreviazione di binary digit, che significa 'cifra binaria'.

La relazione esistente tra i bit e i numeri del sistema decimale è molto semplice:
per convenzione, vengono posti degli zeri prima di ogni numero binario sino a occupare tutte le posizioni disponibili nella notazione corrente. Ci spieghiamo, il numero decimale 7 equivale a 111 binario in notazione su 3 bit; usando una notazione su 5 bit, l'equivalente di 7 in base due diventa 00111.

Cerchiamo ora di capire come avviene la conversione di un numero da decimale in binario. Consideriamo per comodità la notazione binaria su 8 bit, quella utilizzata dalla maggioranza dei computer. In tale notazione ogni numero binario è composto di 8 cifre, ognuna delle quali ha un valore legato alla sua posizione, come avviene nel sistema in base 10. Nel sistema binario il valore è però, ovviamente, relativo alle potenze del numero 2 e non a quelle del numero 10. La posizione meno significativa del numero, quella dell'estrema destra, ha valore 2↑0, cioè 1; la più significativa, quella d'estrema sinistra, ha valore 2↑7, cioè 128. Il numero decimale 7 viene quindi rappresentato nel sistema binario come:

2↑7	2↑6	2↑5	2↑4	2↑3	2↑2	2↑1	2↑0
0	0	0	0	0	1	1	1

infatti:

2↑ + 2↑1 + 2↑2 = 1 + 2 + 4 = 7.

Le cifre binarie sono semplicemente degli indicatori che 'dicono' se considerare (1) o meno (0) nel calcolo del risultato finale la potenza del numero 2 relativa alla posizione in cui si trovano.

Un gruppo di 8 bit viene chiamato byte. È facile stabilire che il massimo numero memorizzabile in un byte è 255, infatti ponendo un 1 in ogni posizione disponibile, si ottiene:

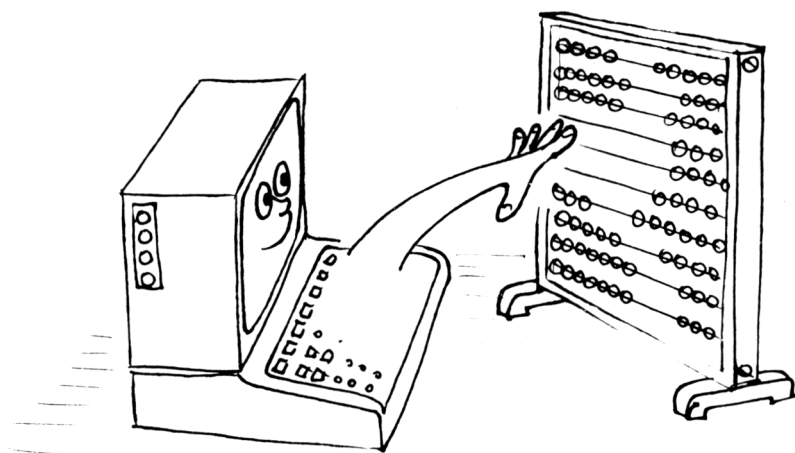
2↑7	2↑6	2↑5	2↑4	2↑3	2↑2	2↑1	2↑0
1	1	1	1	1	1	1	1

che equivale a 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255. Questo implica che un byte può rappresentare 256 combinazioni differenti di 0 e 1. In realtà non si tratta di un numero molto elevato, tant'è che i computer manipolano i dati in multipli di 8 bit. Questo avviene per esempio nell'indirizzamento della memoria, dove vengono utilizzati due byte per indicare una singola cella, tramite la specificazione della sua posizione verticale e orizzontale. Una memoria può essere infatti vista come una matrice:

0	1	2	3	4	5	6	7	8	9
1									
2									
3									
4									
5			1		1				
6									
7									
8									
9									

Questa matrice può contenere sino a 81 elementi utilizzando come indirizzi i numeri da 1 a 9. L'elemento memorizzato alla posizione (4,5) è un '1', così come quello alla posizione (6,5).

Una matrice che utilizzi un byte per l'indirizzo di riga e uno per quello di colonna può quindi contenere 256 × 256 elementi. Ecco come con sole due cifre, 0 e 1, si riesce a identificare univocamente un elemento fra 65536 sfruttando tutte le possibili combinazioni.

Un'altra delle abbreviazioni frequentemente usate quando si parla di dati binari è il Kilobyte, a volte Kbyte o semplicemente K, che vale 1024 byte. Questo è il motivo per cui i computer con '64K' hanno in realtà una memoria di 65536 byte (64 × 1024).

Fortunatamente, il BASIC esegue tutte le conversioni necessarie ed è possibile divenire un buon programmatore senza avere una perfetta conoscenza del sistema di numerazione binario. Comunque, la comprensione del sistema binario è sicuramente d'aiuto per interpretare alcuni dei 'numeri magici' che inevitabilmente si incontrano nel mondo dei computer.

Vale certamente la pena di fare qualche sforzo per comprendere la logica del sistema binario e il significato di numeri come 255, 1024 ecc.. È infatti molto difficile che nel prossimo futuro tali concetti basilari cambino e vengano scavalcati dallo sviluppo tecnologico. La semplicità e la sicurezza che derivano dall'operare con due soli stati prevarrà sulla notevole complessità risultante dall'utilizzo di sistemi di numerazioni diversi.

Comunque......

La semplicità e l'eleganza della notazione binaria si paga però con l'elevato numero di cifre richieste per rappresentare grandi numeri: questo porta all'imprecisione e causa difficoltà nella lettura di un numero binario. Un sistema di numerazione che elimina alcuni di questi difetti è quello esadecimale, con base 16, che utilizza le cifre dallo 0 al 9 e le lettere dalla A alla F. Ecco una prima tabella di conversione da decimale a esadecimale:

Decimale															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Esadecimale															
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Utilizzando la notazione esadecimale è possibile 'spezzare' ogni byte in due gruppi di 4 bit, cioè un semibyte, in quanto il numero 15 (F in esadecimale) può essere rappresentato su 4 bit: 1111 in binario. Il primo semibyte indica il numero

di unità di '15', il secondo il resto, ed è a questo punto che inizia a emergere l'eleganza del sistema binario.

Il numero binario 11010110 equivale a D6 in esadecimale, come potete verificare dividendolo in due semibyte e seguendo la tabella riportata qui di seguito:

Decimale	Binario	Esadecimale
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	10000	10

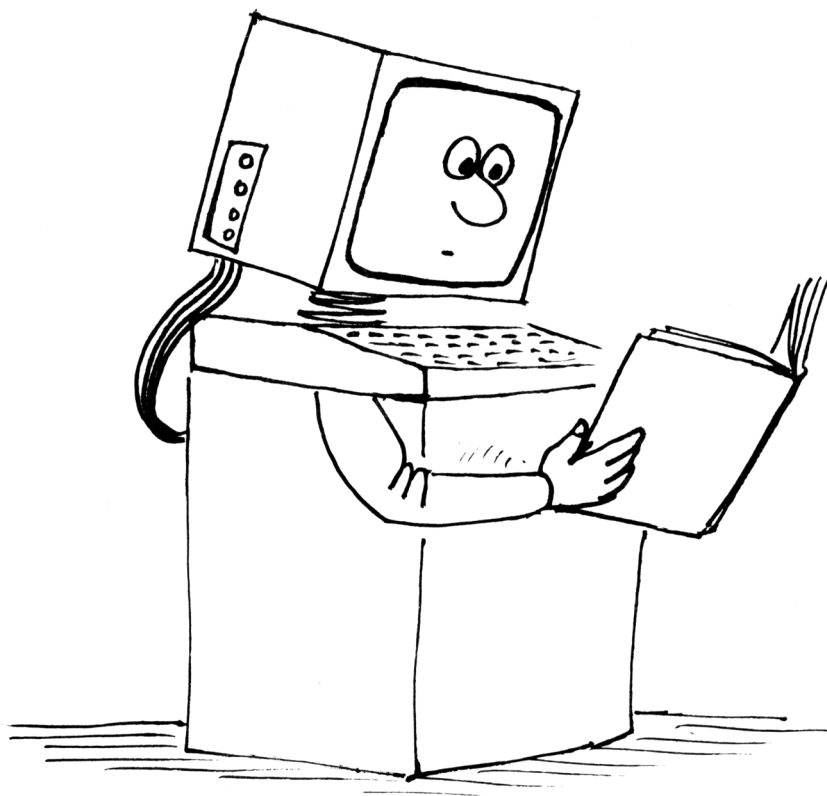
In questa Guida i numeri in base 16 vengono preceduti dal simbolo '&'. Il sistema esadecimale è il più diffuso fra i programmatori che utilizzano il linguaggio Assembler. Questo permette di programmare in linguaggio Macchina utilizzando codici mnemonici composti di lettere e simboli al posto di quelli numerici.

Per calcolare l'equivalente decimale di un numero esadecimale è sufficiente moltiplicare per 16 il valore della prima cifra e poi aggiungere quello della seconda. Quindi il numero &D6 equivale a $(13\times16)+(6) = 214$ in decimale, e non a $13+6$ o a 136 come si sarebbe portati a credere.

È lo stesso procedimento utilizzato con i numeri del sistema decimale, ma è necessario acquisire notevole familiarità con il sistema esadecimale prima che questo divenga di facile uso.

Se fino ad ora non avete avuto eccessivi problemi di comprensione, allora siete sulla strada buona per impadronirvi dei concetti basilari del mondo dei computer. Potreste anche chiedervi, non a torto, il perchè di tutto questo lavoro. Un computer è un oggetto che opera su concetti e idee molto semplici: è in grado di compiere lavori ripetitivi ad elevata velocità (milioni di volte in un secondo) e

possiede una enorme capacità di memorizzare i dati in esso inseriti e tutti i risultati intermedi delle numerosissime operazioni necessarie per ottenere un qualsiasi risultato.

Nel caso siate particolarmente interessati alla teoria dell'informatica potete acquistare uno dei moltissimi libri disponibili sull'argomento 'computer'. Alcuni di questi vi lasceranno idee più confuse di quelle che avevate prima di iniziarne la lettura, solo pochi vi riveleranno la semplicità dei sistemi di numerazione, le relazioni fondamentali esistenti fra di essi e il modo in cui un computer opera con essi.

Appendice III

Il Set di caratteri ASCII e i caratteri grafici del CPC 464

III.1 Il Set ASCII

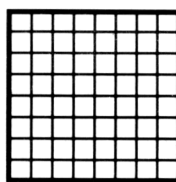
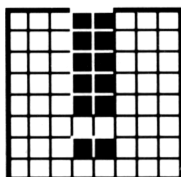
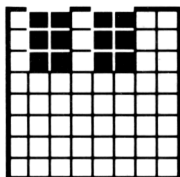
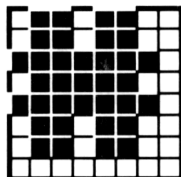
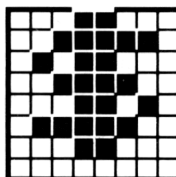
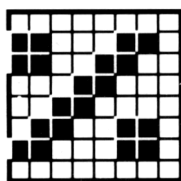
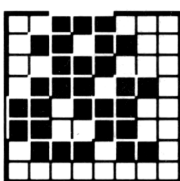
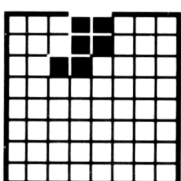
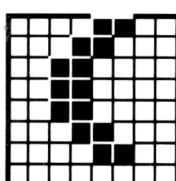
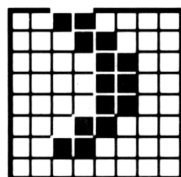
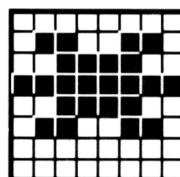
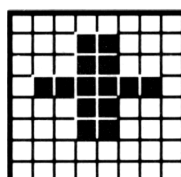
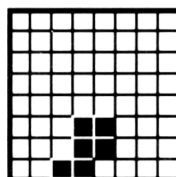
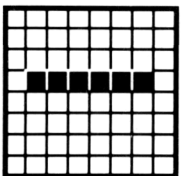
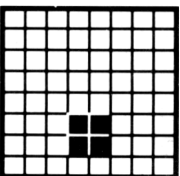
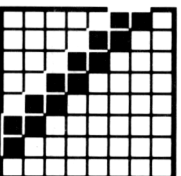
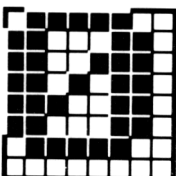
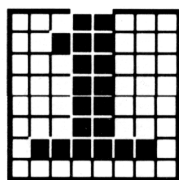
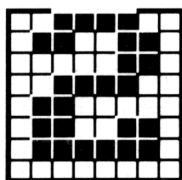
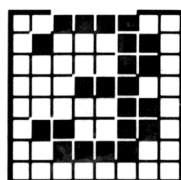
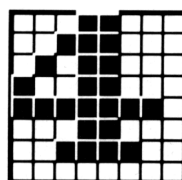
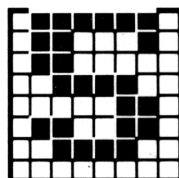
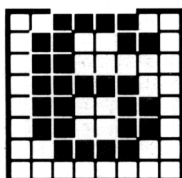
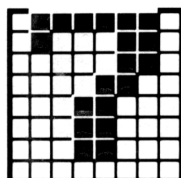
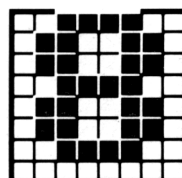
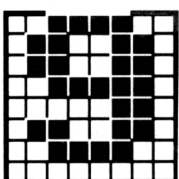
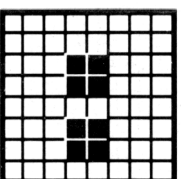
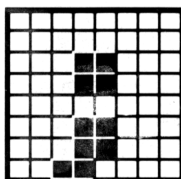
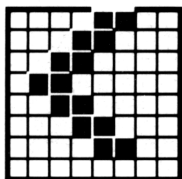
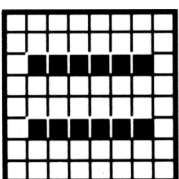
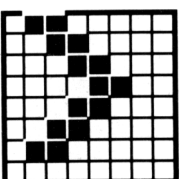
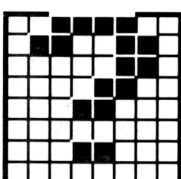
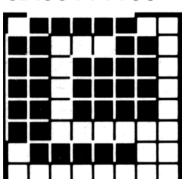
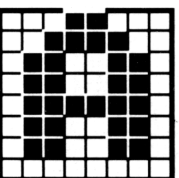
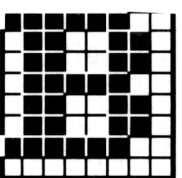
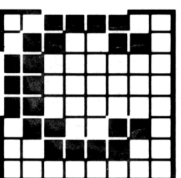
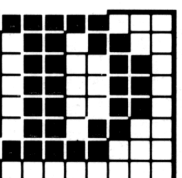
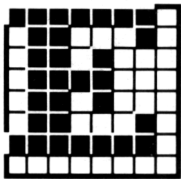
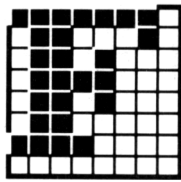
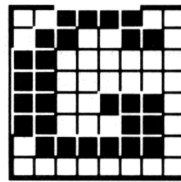
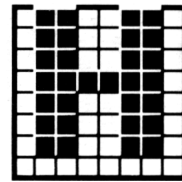
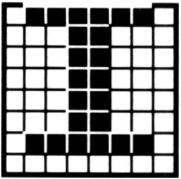
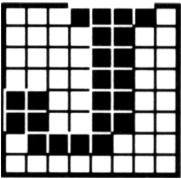
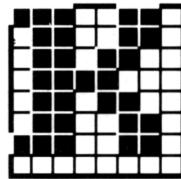
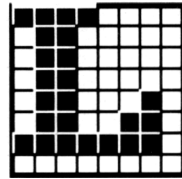
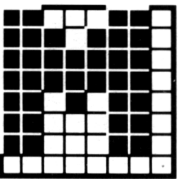
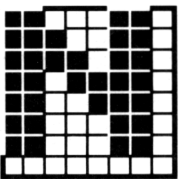
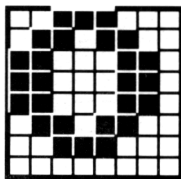
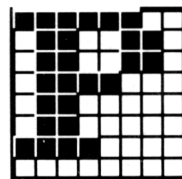
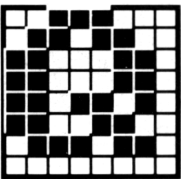
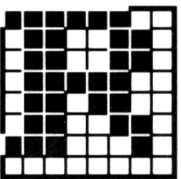
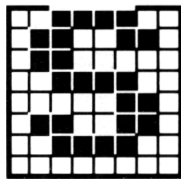
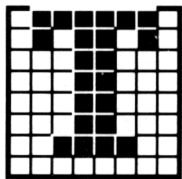
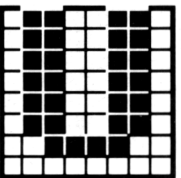
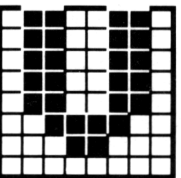
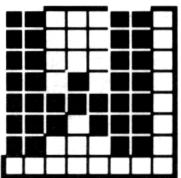
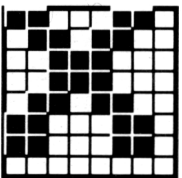
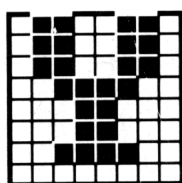
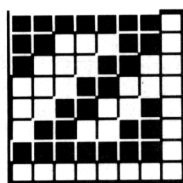
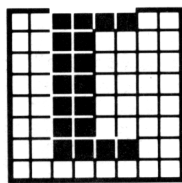
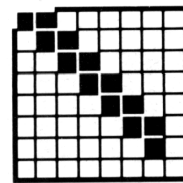
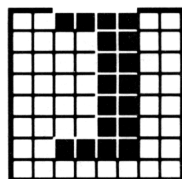
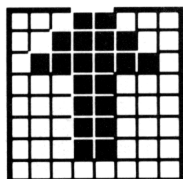
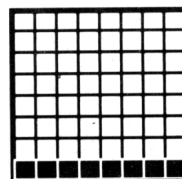
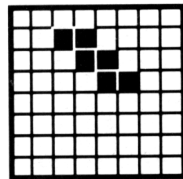
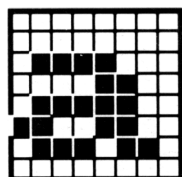
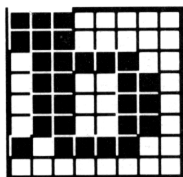
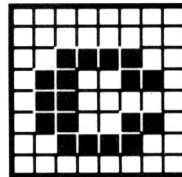
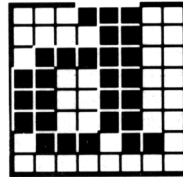
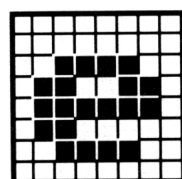
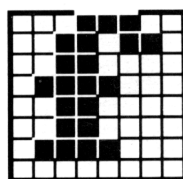
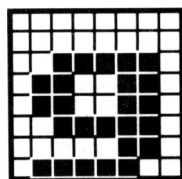
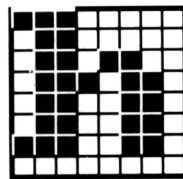
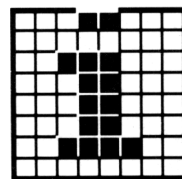
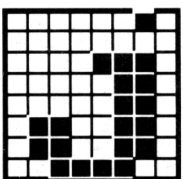
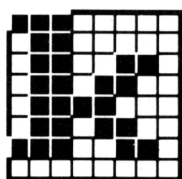
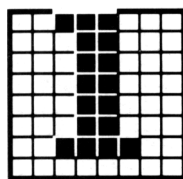
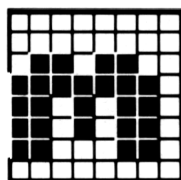
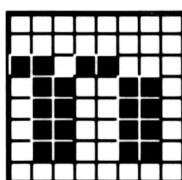
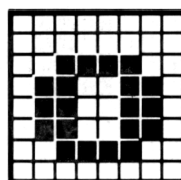
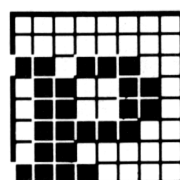
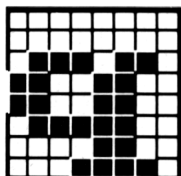
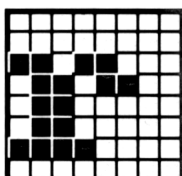
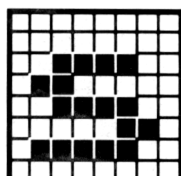
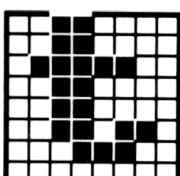
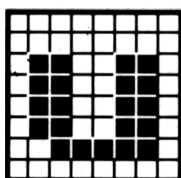
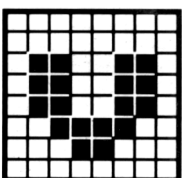
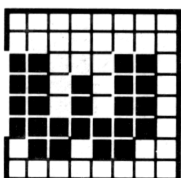
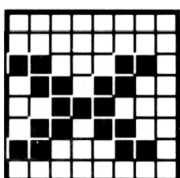
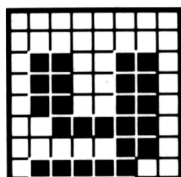
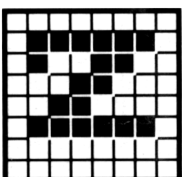
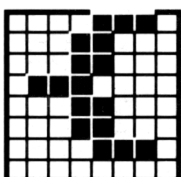
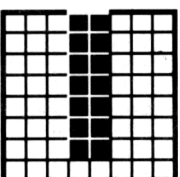
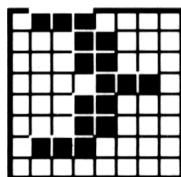
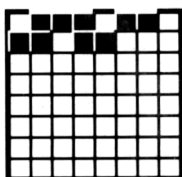
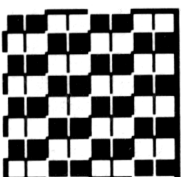
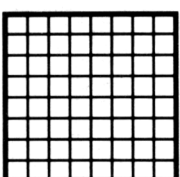
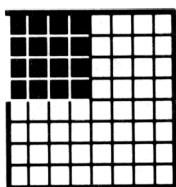
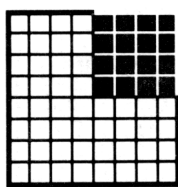
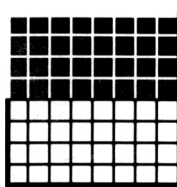
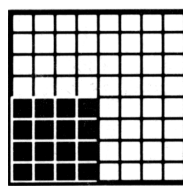
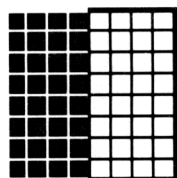
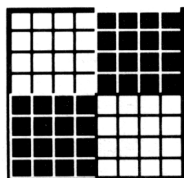
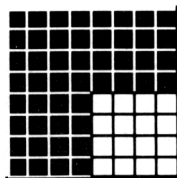
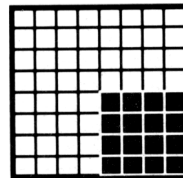
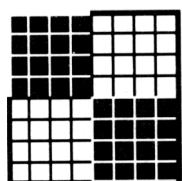
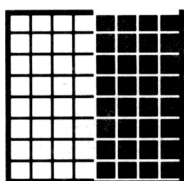
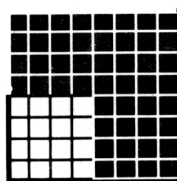
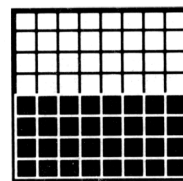
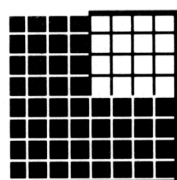
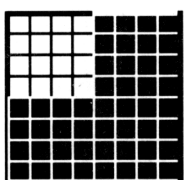
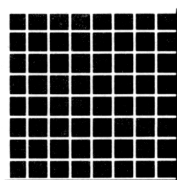
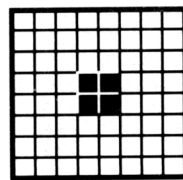
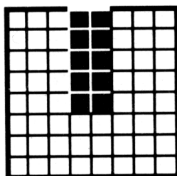
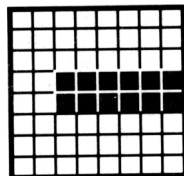
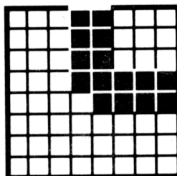
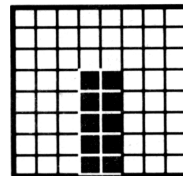
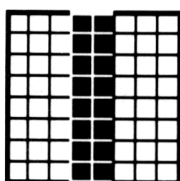
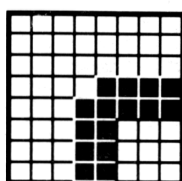
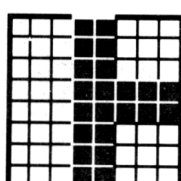
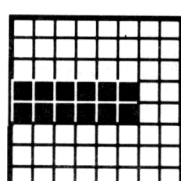
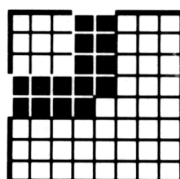
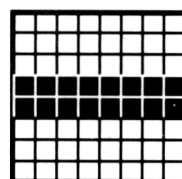
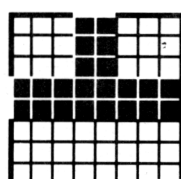
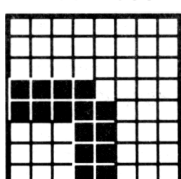
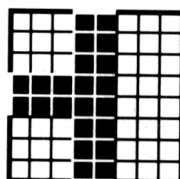
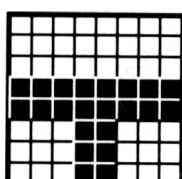
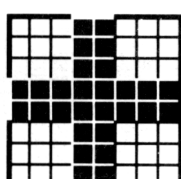
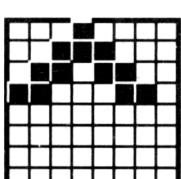
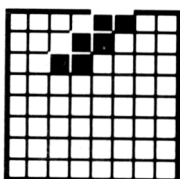
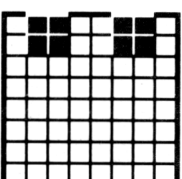
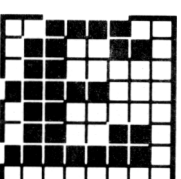
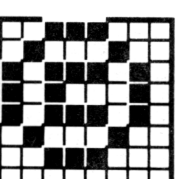
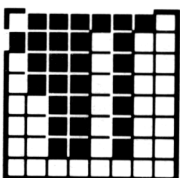
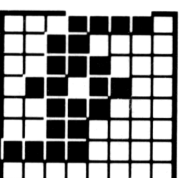
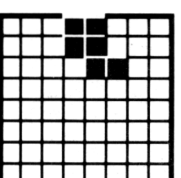
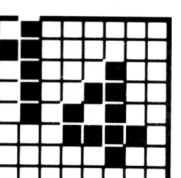
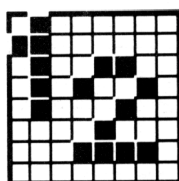
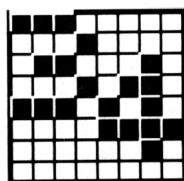
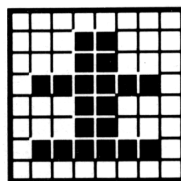
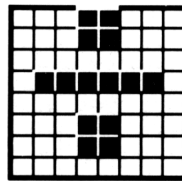
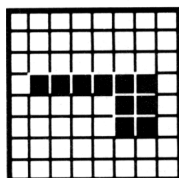
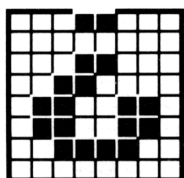
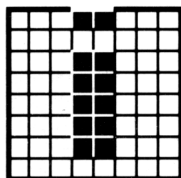
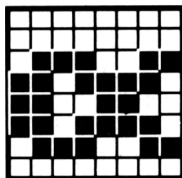
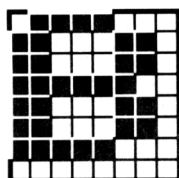
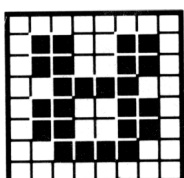
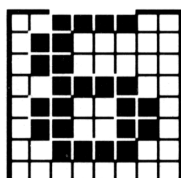
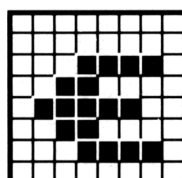
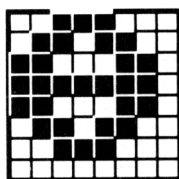
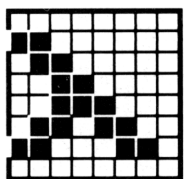
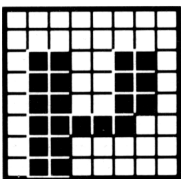
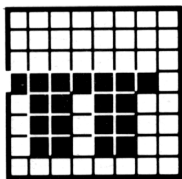
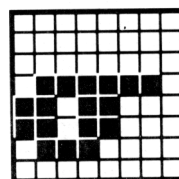
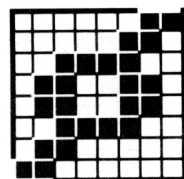
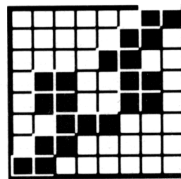
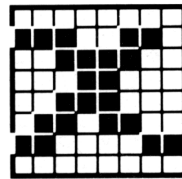
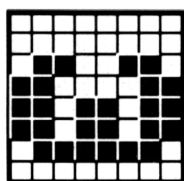
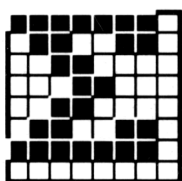
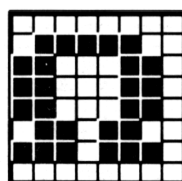
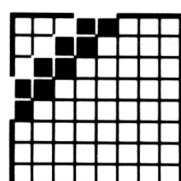
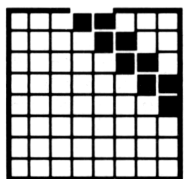
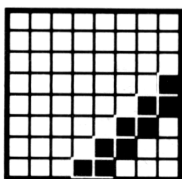
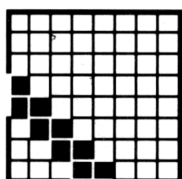
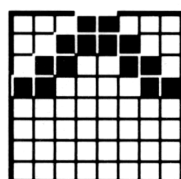
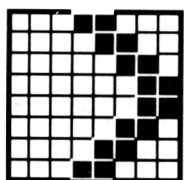
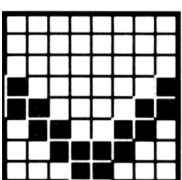
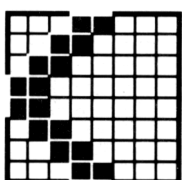
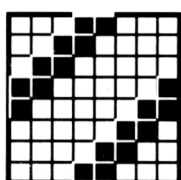
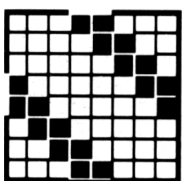
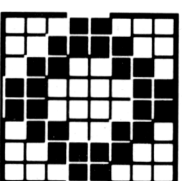
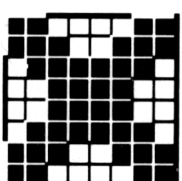
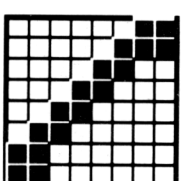
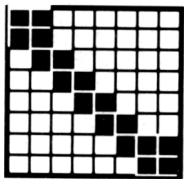
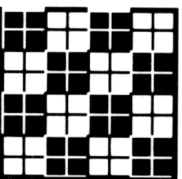
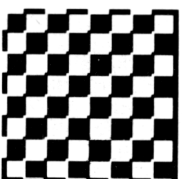
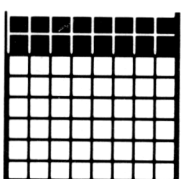
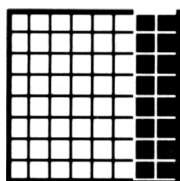
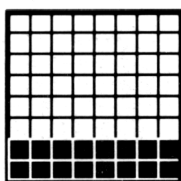
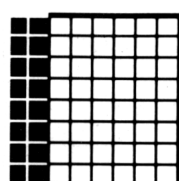
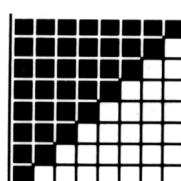
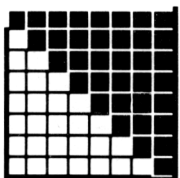
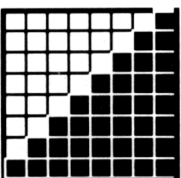
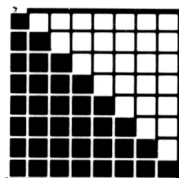
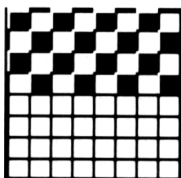
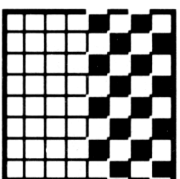
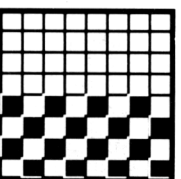
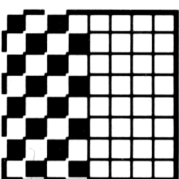
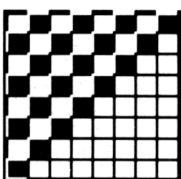
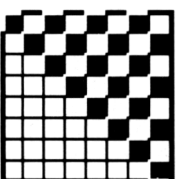
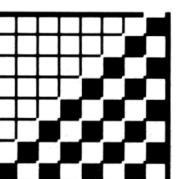
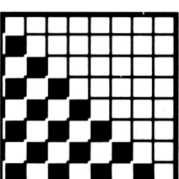
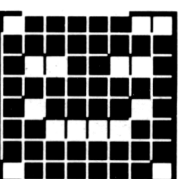
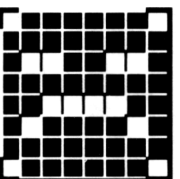
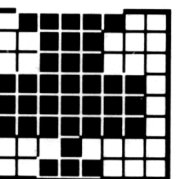
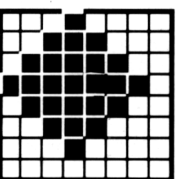
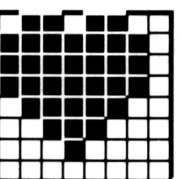
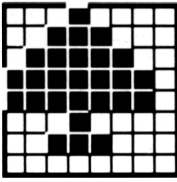
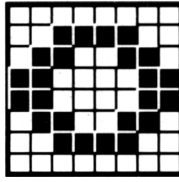
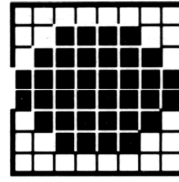
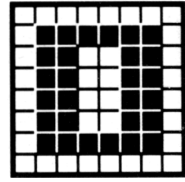
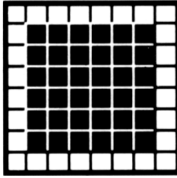
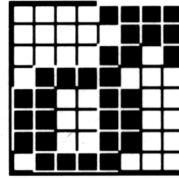
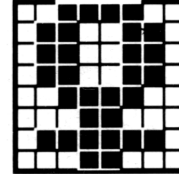
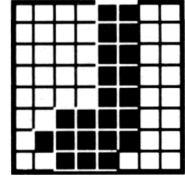
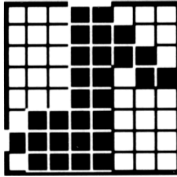
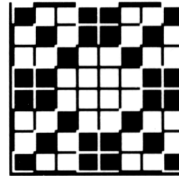
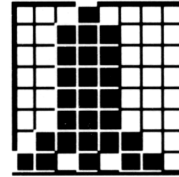
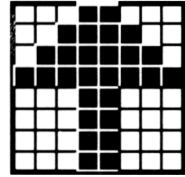
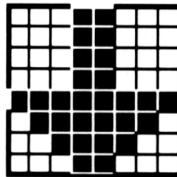
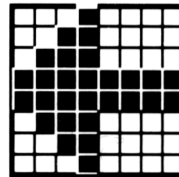
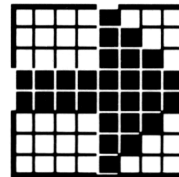
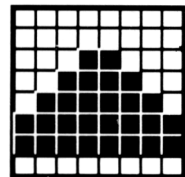
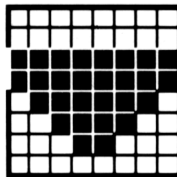
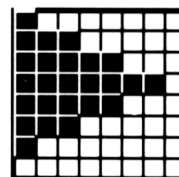
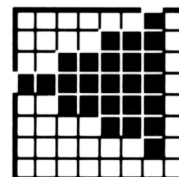
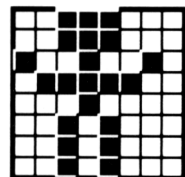
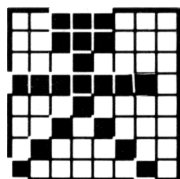
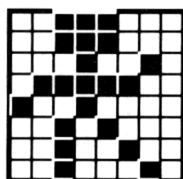
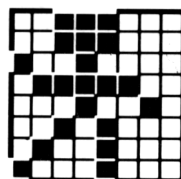
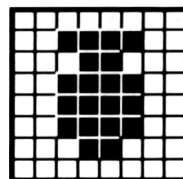
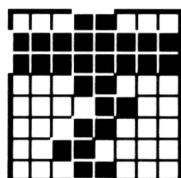
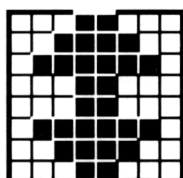
Riportiamo di seguito l'intero Set di caratteri secondo lo standard ASCII. Il codice di ogni carattere è dato in notazione decimale, ottale ed esadecimale.

DEC	OCTAL	HEX	ASCII characters
0	000	00	NUL ([CTRL]@)
1	001	01	SOH ([CTRL]A)
2	002	02	STX ([CTRL]B)
3	003	03	ETX ([CTRL]C)
4	004	04	EOT ([CTRL]D)
5	005	05	ENQ ([CTRL]E)
6	006	06	ACK ([CTRL]F)
7	007	07	BEL ([CTRL]G)
8	010	08	BS ([CTRL]H)
9	011	09	HT ([CTRL]I)
10	012	OA	LF ([CTRL]J)
11	013	OB	VT ([CTRL]K)
12	014	OC	FF ([CTRL]L)
13	015	OD	CR ([CTRL]M)
14	016	OE	SO ([CTRL]N)
15	017	OF	SI ([CTRL]O)
16	020	10	DLE ([CTRL]P)
17	021	11	DC1 ([CTRL]Q)
18	022	12	DC2 ([CTRL]R)
19	023	13	DC3 ([CTRL]S)
20	024	14	DC4 ([CTRL]T)
21	025	15	NAK ([CTRL]U)
22	026	16	SYN ([CTRL]V)
23	027	17	ETB ([CTRL]W)
24	030	18	CAN ([CTRL]X)
25	031	19	EM ([CTRL]Y)
26	032	1A	SUB ([CTRL]Z)
27	033	1B	ESC
28	034	1C	FS
29	035	1D	GS
30	036	1E	RS
31	037	1F	US
32	040	20	SP
33	041	21	!
34	042	22	"
35	043	23	#
36	044	24	$
37	045	25	%
38	046	26	&
39	047	27	'
40	050	28	(
41	051	29	)
42	052	2A	*
43	053	2B	+
44	054	2C	,
45	055	2D	-
46	056	2E	.
47	057	2F	/
48	060	30	Ø
49	061	31	1

DEC	OCTAL	HEX	ASCII
50	062	32	2
51	063	33	3
52	064	34	4
53	065	35	5
54	066	36	6
55	067	37	7
56	070	38	8
57	071	39	9
58	072	3A	:
59	073	3B	;
60	074	3C	<
61	075	3D	=
62	076	3E	>
63	077	3F	?
64	100	40	@
65	101	41	A
66	102	42	B
67	103	43	C
68	104	44	D
69	105	45	E
70	106	46	F
71	107	47	G
72	110	48	H
73	111	49	I
74	112	4A	J
75	113	4B	K
76	114	4C	L
77	115	4D	M
78	116	4E	N
79	117	4F	O
80	120	50	P
81	121	51	Q
82	122	52	R
83	123	53	S
84	124	54	T
85	125	55	U
86	126	56	V
87	127	57	W
88	130	58	X
89	131	59	Y
90	132	5A	Z
91	133	5B	[
92	134	5C	\
93	135	5D	]
94	136	5E	↑
95	137	5F	–
96	140	60	`
97	141	61	a
98	142	62	b
99	143	63	c

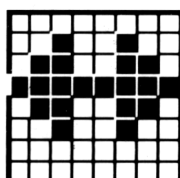
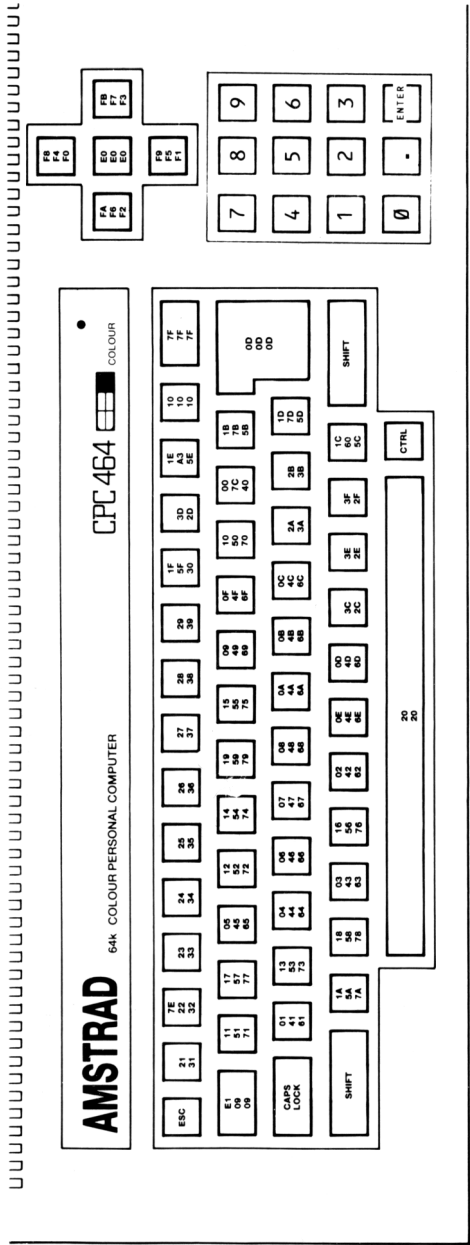
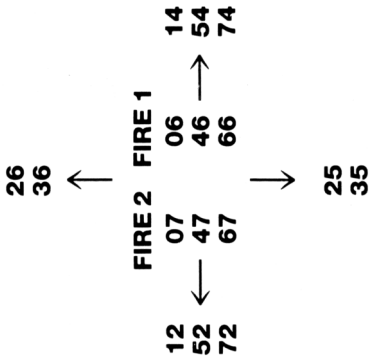
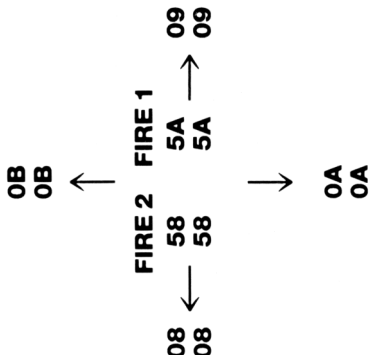
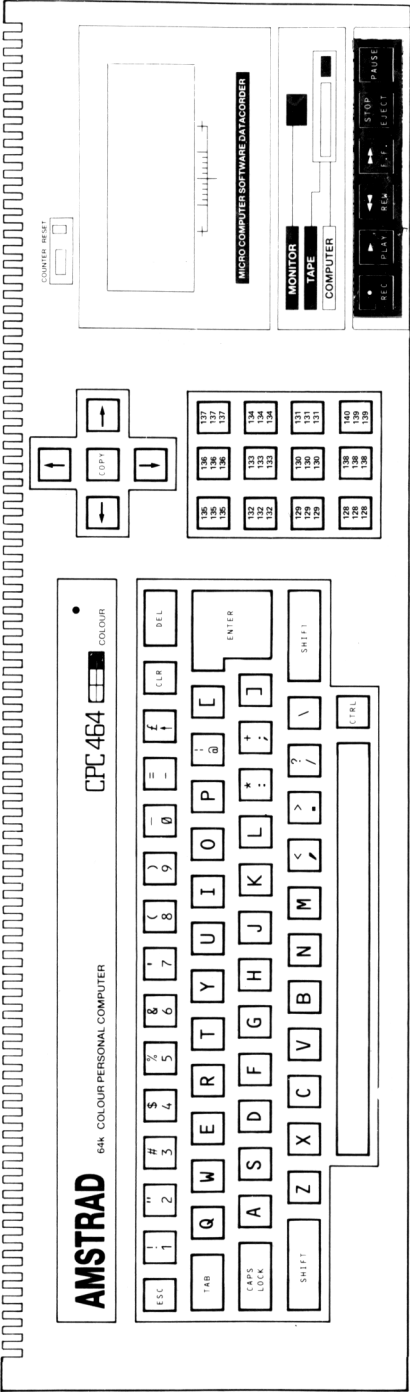
DEC	OCTAL	HEX	ASCII
100	144	64	d
101	145	65	e
102	146	66	f
103	147	67	g
104	150	68	h
105	151	69	i
106	152	6A	j
107	153	6B	k
108	154	6C	l
109	155	6D	m
110	156	6E	n
111	157	6F	o
112	160	70	p
113	161	71	q
114	162	72	r
115	163	73	s
116	164	74	t
117	165	75	u
118	166	76	v
119	167	77	w
120	170	78	x
121	171	79	y
122	172	7A	z
123	173	7B	{
124	174	7C	|
125	175	7D	}
126	176	7E	~

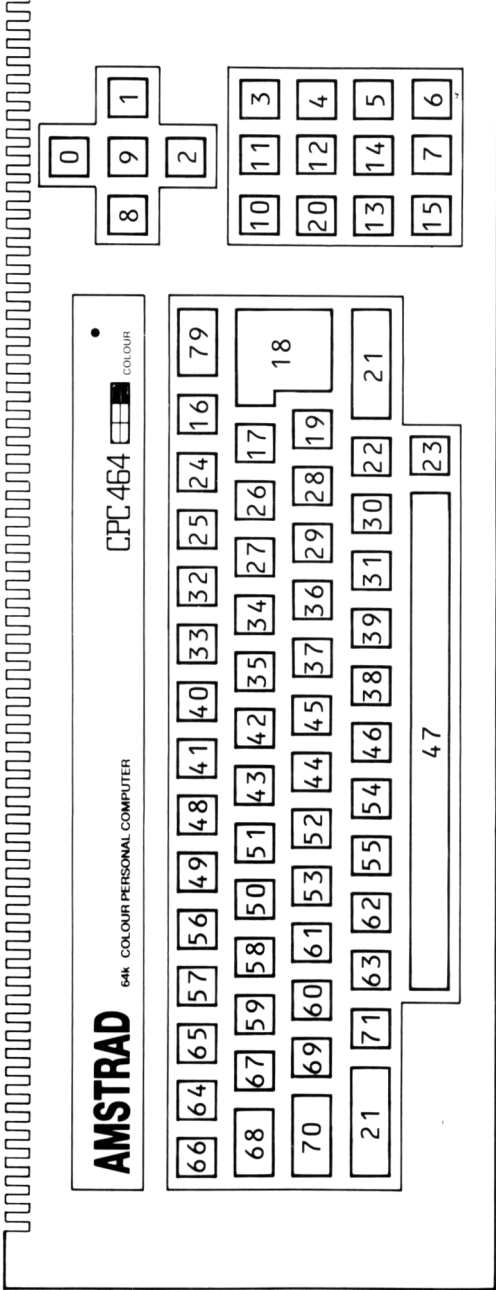
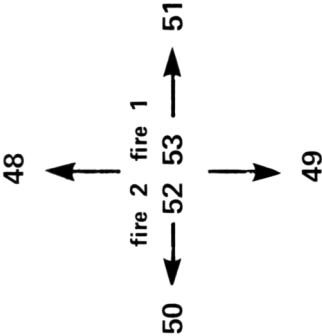
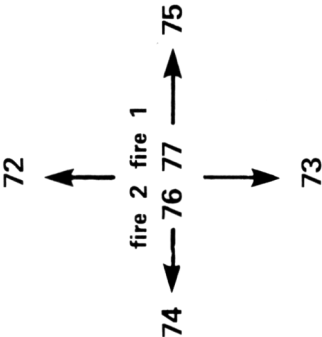
III.2 Il Set di caratteri del CPC 464

Riportiamo di seguito la matrice di tutti i caratteri del CPC 464, che vengono stampati sullo schermo in una griglia di 8×8 pixel. Il comando SYMBOL, la cui descrizione si trova nel Capitolo 8, permette di definire nuove matrici di carattere accostabili fra loro per ottenere qualsiasi simbolo.

32 &H20
&X00100000

33
&H21
&X00100001

34
&H22
&X00100010

35
&H23
&X00100011

36
&H24
&X00100100

37
&H25
&X00100101

38
&H26
&X00100110

39
&H27
&X00100111

40
&H28
&X00101000

41
&H29
&X00101001

42
&H2A
&X00101010

43
&H2B
&X00101011

44
&H2C
&X00101100

45
&H2D
&X00101101

46
&H2E
&X00101110

47
&H2F
&X00101111

48
&H30
&X00110000

49
&H31
&X00110001

50
&H32
&X00110010

51
&H33
&X00110011

52
&H34
&X00110100

53
&H35
&X00110101

54
&H36
&X00110110

55
&H37
&X00110111

56
&H38
&X00111000

57
&H39
&X00111001

58
&H3A
&X00111010

59
&H3B
&X00111011

60
&H3C
&X00111100

61
&H3D
&X00111101

62
&H3E
&X00111110

63
&H3F
&X00111111

64
&H40
&X01000000

65
&H41
&X01000001

66
&H42
&X01000010

67
&H43
&X01000011

68
&H44
&X01000100

69
&H45
&X01000101

70
&H46
&X01000110

71
&H47
&X01000111

72
&H48
&X01001000

73
&H49
&X01001001

74
&H4A
&X01001010

75
&H4B
&X01001011

76
&H4C
&X01001100

77
&H4D
&X01001101

78
&H4E
&X01001110

79
&H4F
&X01001111

80
&H50
&X01010000

81
&H51
&X01010001

82
&H52
&X01010010

83
&H53
&X01010011

84
&H54
&X01010100

85
&H55
&X01010101

86
&H56
&X01010110

87
&H57
&X01010111

88
&H58
&X01011000

89 &H59 &X01011001

90 &H5A &X01011010

91 &H5B &X01011011

92 &H5C &X01011100

93 &H5D &X01011101

94 &H5E &X01011110

95 &H5F &X01011111

96 &H60 &X01100000

97 &H61 &X01100001

98 &H62 &X01100010

99 &H63 &X01100011

100 &H64 &X01100100

101 &H65 &X01100101

102 &H66 &X01100110

103 &H67 &X01100111

104 &H68 &X01101000

105 &H69 &X01101001

106 &H6A &X01101010

107 &H6B &X01101011

108 &H6C &X01101100

109
&H6D
&X01101101

110
&H6E
&X01101110

111
&H6F
&X01101111

112
&H70
&X01110000

113
&H71
&X01110001

114
&H72
&X01110010

115
&H73
&X01110011

116
&H74
&X01110100

117
&H75
&X01110101

118
&H76
&X01110110

119
&H77
&X01110111

120
&H78
&X01111000

121
&H79
&X01111001

122
&H7A
&X01111010

123
&H7B
&X01111011

124
&H7C
&X01111100

125
&H7D
&X01111101

126
&H7E
&X01111110

127
&H7F
&X01111111

128
&H80
&X10000000

129
&H81
&X10000001

130
&H82
&X10000010

131
&H83
&X10000011

132
&H84
&X10000100

133
&H85
&X10000101

134
&H86
&X10000110

135
&H87
&X10000111

136
&H88
&X10001000

137
&H89
&X10001001

138
&H8A
&X10001010

139
&H8B
&X10001011

140
&H8C
&X10001100

141
&H8D
&X10001101

142
&H8E
&X10001110

143
&H8F
&X10001111

144
&H90
&X10010000

145
&H91
&X10010001

146
&H92
&X10010010

147
&H93
&X10010011

148
&H94
&X10010100

149 &H95 &X10010101

150 &H96 &X10010110

151 &H97 &X10010111

152 &H98 &X10011000

153 &H99 &X10011001

154 &H9A &X10011010

155 &H9B &X10011011

156 &H9C &X10011100

157 &H9D &X10011101

158 &H9E &X10011110

159 &H9F &X10011111

160 &HA0 &X10100000

161 &HA1 &X10100001

162 &HA2 &X10100010

163 &HA3 &X10100011

164 &HA4 &X10100100

165 &HA5 &X10100101

166 &HA6 &X10100110

167 &HA7 &X10100111

168 &HA8 &X10101000

169
&HA9
&X10101001

170
&HAA
&X10101010

171
&HAB
&X10101011

172
&HAC
&X10101100

173
&HAD
&X10101101

174
&HAE
&X10101110

175
&HAF
&X10101111

176
&HB0
&X10110000

177
&HB1
&X10110001

178
&HB2
&X10110010

179
&HB3
&X10110011

180
&HB4
&X10110100

181
&HB5
&X10110101

182
&HB6
&X10110110

183
&HB7
&X10110111

184
&HB8
&X10111000

185
&HB9
&X10111001

186
&HBA
&X10111010

187
&HBB
&X10111011

188
&HBC
&X10111100

189
&HBD
&X10111101
190
&HBE
&X10111110
191
&HBF
&X10111111
192
&HC0
&X11000000
193
&HC1
&X11000001
194
&HC2
&X11000010
195
&HC3
&X11000011
196
&HC4
&X11000100
197
&HC5
&X11000101
198
&HC6
&X11000110
199
&HC7
&X11000111
200
&HC8
&X11001000
201
&HC9
&X11001001
202
&HCA
&X11001010
203
&HCB
&X11001011
204
&HCC
&X11001100
205
&HCD
&X11001101
206
&HCE
&X11001110
207
&HCF
&X11001111
208
&HD0
&X11010000

209
&HD1
&X11010001
210
&HD2
&X11010010
211
&HD3
&X11010011
212
&HD4
&X11010100
213
&HD5
&X11010101
214
&HD6
&X11010110
215
&HD7
&X11010111
216
&HD8
&X11011000
217
&HD9
&X11011001
218
&HDA
&X11011010
219
&HDB
&X11011011
220
&HDC
&X11011100
221
&HDD
&X11011101
222
&HDE
&X11011110
223
&HDF
&X11011111
224
&HE0
&X11100000
225
&HE1
&X11100001
226
&HE2
&X11100010
227
&HE3
&X11100011
228
&HE4
&X11100100

229
&HE5
&X11100101
230
&HE6
&X11100110
231
&HE7
&X11100111
232
&HE8
&X11101000
233
&HE9
&X11101001
234
&HEA
&X11101010
235
&HEB
&X11101011
236
&HEC
&X11101100
237
&HED
&X11101101
238
&HEE
&X11101110
239
&HEF
&X11101111
240
&HF0
&X11110000
241
&HF1
&X11110001
242
&HF2
&X11110010
243
&HF3
&X11110011
244
&HF4
&X11110100
245
&HF5
&X11110101
246
&HF6
&X11110110
247
&HF7
&X11110111
248
&HF8
&X11111000

249
&HF9
&X11111001

250
&HFA
&X11111010

251
&HFB
&X11111011

252
&HFC
&X11111100

253
&HFD
&X11111101

254
&HFE
&X11111110

255
&HFF
&X11111111

Pagina III.14 — Valori ASCII assegnati alla tastiera

Pagina III.15 — Tasti definibili dall'utente, codici di default

exp char	value	ascii
128	0	30
129	1	31
130	2	32
131	3	33
132	4	34
133	5	35
134	6	36
135	7	37
136	8	38
137	9	39
138		2E
139	[enter]	0D
140	run"	52,55,4E,22,0D

Pagina III.16 – Codici dei tasti e dei joystick

Appendice IV

GUIDA ALL'USO DEL CPC 464 PER L'UTENTE ESPERTO

L'hardware e il firmware del CPC 464 sono stati progettati e costruiti per offrire, sia al principiante che all'utente esperto, la possibilità di utilizzare programmi già esistenti o di scriverne di nuovi, sempre sfruttando al massimo le notevoli capacità del computer.

Le principali caratteristiche del sistema sono:

CPU Z80A

È il microprocessore ad 8 bit più utilizzato dai costruttori di home computer e per il quale è disponibile il miglior software. In particolar modo, la possibilità di implementare il noto Sistema Operativo CP/M permette ai computer basati su Z80 di accedere ad una ricchissima biblioteca di programmi già ampiamente collaudati. La struttura di gestione degli interrupt direttamente da BASIC, unica per potenza e flessibilità, mette a disposizione degli utenti prestazioni innovative, quali quelle dei comandi AFTER e EVERY.

64K RAM

Grazie alla particolare organizzazione del sistema operativo, in grado di gestire la memoria a 'pagine', oltre 42K di memoria RAM sono disponibili per l'utente, nonostante l'interprete BASIC occupi 32K di ROM.

Schermo

Il CPC 464 dispone di tre modi di organizzazione dello schermo, che gli consentono prestazioni uniche, quali la possibilità di visualizzare 80 caratteri per riga in modo testo o di indirizzare 640 × 200 pixel in alta risoluzione. Non di meno, la tavolozza dei colori comprende ben 27 tinte tra cui scegliere.

Tastiera professionale

La tastiera, standard QWERTY, include un pad numerico separato, i cui tasti sono ridefinibili come tasti funzione.

Registratore di cassette incorporato

Il registratore di cassette incorporato garantisce una elevata affidabilità nelle operazioni su nastro ed elimina i problemi di connessione o di regolazione dei livelli di volume e di tono. La velocità di salvataggio dei programmi, selezionabile da BASIC, può essere di 1K o 2K baud; quella di lettura viene automaticamente stabilita dal CPC 464.

BASIC

L'interprete BASIC, scritto in Inghilterra dalla Locomotive, è eccezionalmente completo, veloce ed esteso alla gestione della grafica, del suono e del colore.

Set di caratteri esteso

È disponibile un set di 256 caratteri, accessibili sia da tastiera che tramite il comando BASIC CHR$(n). L'intero set di caratteri può essere ridefinito.

Temporizzazione

L'utente ha accesso, direttamente da BASIC, a quattro timer interni indipendenti l'uno dall'altro.

Tasti definibili

Esiste la possibilità di ridefinire fino a 32 tasti, assegnando ad ognuno una stringa di lunghezza massima pari a 32 caratteri. Inoltre, anche la velocità di ripetizione dei tasti può essere modificata a piacere.

Subroutine

Molte subroutine del Sistema Operativo possono essere richiamate da BASIC.

Modi di organizzazione dello schermo

Sono disponibili tre Modi:

a) Normale

Modo 1: 40 colonne × 25 righe, 4 colori per il testo
320 × 200 pixel, ognuno singolarmente indirizzabile con 4 INK

b) Modo Multicolore

Modo 0: 20 colonne × 25 righe, 16 colori per il testo
160 × 200 pixel, ognuno singolarmente indirizzabile con 16 INK

c) Alta risoluzione

Modo 2: 80 colonne × 25 righe, 2 colori per il testo
640 × 200 pixel, ognuno singolarmente indirizzabile con 2 INK

Selezione dei colori

(NB Nel prosieguo di questa guida il 'nero' — cioè l'assenza di colore — è considerato a tutti gli effetti come un colore)

Per il bordo è possibile selezionare una coppia qualsiasi di colori, indipendentemente dal Modo in cui ci si trova. Assegnando al bordo due colori, gli stessi verranno visualizzati in alternanza fra loro, producendo un effetto di flashing.

Il numero di colori contemporaneamente visualizzabili ed il numero di inchiostri (INK) disponibili dipendono dal Modo selezionato. Ad ogni INK è possibile associare una coppia qualsiasi di colori in modo da ottenere il lampeggio. Il colore di fondo dei caratteri (PAPER), quello con cui vengono stampati (PEN) e quello della Penna Grafica possono essere associati ad uno qualsiasi degli inchiostri disponibili.

La visualizzazione del testo può essere effettuata in modo 'opaco' o 'trasparente', ignorando cioè il colore di fondo e sovrascrivendo la grafica, oppure sovrapponendosi completamente allo sfondo.

Finestre

L'utente può aprire sino ad otto finestre di testo in cui scrivere caratteri e una finestra grafica nella quale disegnare. Cambiando Modo di visualizzazione si provoca la chiusura di tutte le finestre eventualmente aperte.

NB: Se la finestra di testo è in realtà equivalente a tutto lo schermo (come per default), le operazioni di scroll vengono eseguite via hardware, risultando perciò molto veloci. Se invece la finestra di testo è più piccola dell'intero schermo, le stesse operazioni devono essere eseguite dal software di base, risultando più lente.

Cursore

Il cursore, rappresentato da un quadrato in inverse, è disabilitato quando la CPU non è in attesa di un input.

Suono Polifonico

Il CPC 464 genera gli effetti sonori tramite un processore dedicato della famiglia AY8910 prodotta della General Instrument. Tale processore opera su tre canali indipendenti, ognuno dei quali può essere controllato in tono e volume. Ru-

more può essere aggiunto ai suoni a seconda delle necessità. Il suono può essere inviato verso un amplificatore esterno tramite l'apposito jack presente sul retro del computer.

Il BASIC permette di definire gli inviluppi di tono e di volume. Le possibilità di inviluppo di volume proprie del generatore sonoro non vengono normalmente utilizzate.

Porta Stampante

Il CPC 464 dispone di un'interfaccia parallela standard Centronics che utilizza il segnale 'Busy' per effettuare l'handshaking con la periferica.

Possibilità di espansione

Il sistema operativo, così come l'hardware, è già predisposto per il trattamento di periferiche aggiuntive. I driver software per la gestione di dette periferiche devono essere contenuti in ROM esterne connesse al bus di espansione.

Coordinate

Il sistema di riferimento delle coordinate di schermo ha origine in punti diversi a seconda che si tratti di testo o di grafica. Nel primo caso l'origine si trova nell'angolo in alto a sinistra e a coordinate uguali corrispondono diverse posizioni del carattere a seconda del Modo selezionato; in grafica, l'origine è posta nell'angolo in basso a sinistra dello schermo e le coordinate rimangono invariate nei differenti Modi.

NB:

In Modo 1, ad ogni pixel corrispondono due indirizzi, così come in Modo 0 ne corrispondono quattro, e tutti possono essere utilizzati per indirizzare il pixel.

Le coordinate sull'asse verticale vanno da 0 a 399 e vengono divise per due per ottenere l'indirizzo di una delle 200 posizioni fisicamente disponibili. Questo assicura che le figure rappresentate sullo schermo non subiscano deformazioni.

Espansioni di ROM

La ROM viene indirizzata nella parte alta della memoria e il firmware è in grado di gestire sino a 240 ulteriori banchi da 16K di ROM.

Descrizione generale:

Un breve sommario delle principali caratteristiche dell'hardware e del firmware del CPC 464.

4.1 — Hardware

All'interno del contenitore del CPC 464 si trovano: computer, tastiera, registratore Datacorder, altoparlante e l'uscita RGB per il monitor.

4.1.1 — Integrati LSI

Microprocessore Z80A con clock di 4Mhz

64K byte di memoria RAM dinamica; il refresh viene eseguito insieme con l'accesso alla memoria video.

32K byte di ROM contenenti il BASIC e il Sistema Operativo.

Una ULA custom contiene i circuiti per la temporizzazione, la generazione del colore e le operazioni di DMA.

Il CRT controller 6845 genera i segnali per la scansione della memoria video.

NB:

Lo schermo è mappato in maniera complessa e cambia a seconda del Modo di visualizzazione. Il CRT controller può essere utilizzato dal software per eseguire lo scrolling e il rolling della pagina video. I parametri inviati al CRT controller stabiliscono il numero di linee dello schermo, la frequenza di scansione, l'ampiezza e il posizionamento dei bordi.

Il generatore sonoro AY-3-8912 produce suoni a 3 voci. Il suono viene preso dai tre canali e miscelato per ottenere una uscita mono sull'altoparlante interno il cui volume può essere regolato tramite l'apposito controllo presente sul lato destro del CPC 464. Esiste anche un'uscita stereo dove:

Canale sinistro = Canale A + 1/2 Canale C
Canale destro = Canale B + 1/2 Canale C

Questo circuito integrato riceve anche le informazioni provenienti dalla scansione della tastiera e della porta joystick.

L'interfacciamento del generatore sonoro con il bus è affidato ad un 8255, che esegue anche la scansione della tastiera, della porta joystick e controlla il Datacorder.

4.1.2 — Connettori esterni

Sul retro si trovano un connettore PCB general purpose, la porta parallela (Centronics) per la stampante, la presa per il joystick (DIN 9 pin), il jack femmina per inviare il suono ad un amplificatore stereo e l'uscita video RGB.

4.2 — Fuori dal contenitore

Il CPC 464 può essere collegato a due tipi di monitor, a colori o fosfori verdi, che forniscono anche l'alimentazione richiesta dal sistema. Un Modulatore/Alimentatore esterno, l'MP1, può essere utilizzato per collegare il computer con un apparecchio TV standard.

Il collegamento del CPC 464 ad una stampante compatibile Centronics e ad un impianto HI - FI richiederà l'uso di cavi.

4.3 — Specifiche dello schermo

La memoria video è di 16K. La macchina ha un set di 27 colori che possono essere liberamente selezionati dalla tavolozza. Il numero di INK utilizzabili dipende dal Modo in cui ci si trova. Se necessario, lo stesso colore può essere associato a diversi INK. I pixel sullo schermo sono in realtà punti illuminati con un dato colore. Un bordo contorna la zona indirizzabile dello schermo e può essere colorato con uno qualsiasi dei 27 colori disponibili.

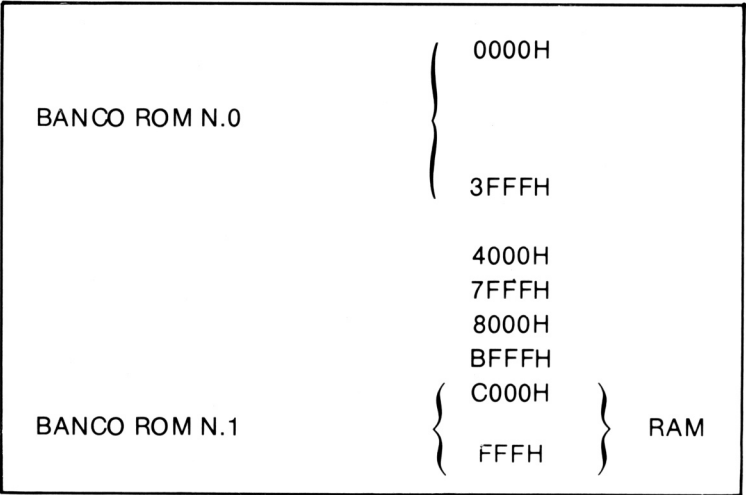
MODO	INK disp.	Pixel Vert.	Pixel Orizz.	Caratteri
Normale	4	200	320	40
Alta Ris.	2	200	640	80
Multicolore	16	200	160	20

NB: Utilizzando il monitor a fosfori verdi i colori corrispondono alla scala dei grigi, in ordine crescente di luminosità, come riportato nella tabella che segue.

LIVELLO DI GRIGIO	COLORE	LIVELLO DI GRIGIO	COLORE
0	NERO	13	BIANCO
1	BLU	14	BLU PASTELLO
2	BLU CHIARO	15	ARANCIONE
3	ROSSO	16	ROSA
4	MAGENTA	17	MAGENTA PASTELLO
5	MALVA	18	VERDE CHIARO
6	ROSSO CHIARO	19	VERDE MARE
7	PORPORA	20	CIANO CHIARO
8	MAGENTA CHIARO	21	VERDE LIME
9	VERDE	22	VERDE PASTELLO
10	CIANO	23	CIANO PASTELLO
11	BLU CIELO	24	GIALLO CHIARO
12	GIALLO	25	GIALLO PASTELLO
		26	BIANCO CHIARO

4.4 — Mappa della memoria

I 64K di RAM del CPC 464 sono allocati come segue.

Notate che una parte della ROM si sovrappone alla memoria video, lasciando così a disposizione del BASIC la massima quantità di memoria.

Nel caso che lo stesso indirizzo identifichi sia una locazione RAM che una locazione ROM, allora le operazioni di LETTURA saranno riferite alla ROM, mentre quelle di SCRITTURA si riferiranno alla RAM. È comunque possibile escludere ciascuna delle due sezioni ROM, in modo da aver accesso alla RAM in lettura anche nel caso di identità di indirizzo con la ROM.

4.5 — Espandibilità

4.5.1 —Banchi ROM addizionali

Esiste la possibilità di selezionare banchi di ROM addizionali in luogo di qualsiasi parte della ROM fornita in standard. L'indirizzamento e la logica di selezione dei banchi saranno su di un modulo connesso al bus di espansione, mentre tutti i segnali richiesti sono presenti su detto bus.

4.5.2 — RAM addizionale

Vale quanto già detto al paragrafo precedente per la ROM. L'eventuale memoria RAM aggiuntiva sarà disponibile solo in lettura, a meno di non mettere a punto dispositivi circuitali speciali per la mappatura dell'I/O.

4.5.3 — I/O addizionale

La maggior parte degli indirizzi delle porte di input/output sono riservati al sistema; in particolare, occorre evitare assolutamente di usare gli indirizzi sotto 7Fxx. Gli indirizzi utilizzabili da hardware esterno sono:

F8xx, F9xx, FAxx, FBxx

I bus delle periferiche di espansione devono decodificare gli indirizzi da A0 a A7 mentre l'indirizzo A10 è nello stato logico zero. I canali di I/O da F800 a FBFF sono riservati come segue:

Indirizzi da A0 a A7

00 — 7B	** non devono essere utilizzati **
7C — 7F	riservati per il disk controller
80 — BB	** non devono essere utilizzati **
BC — BF	riservati per espansioni future
C0 — DB	** non devono essere utilizzati **
DC — DF	riservati per le interfacce di comunicazione
E0 — FF	disponibili per le periferiche dell'utente

Tenete presente che per le operazioni di I/O devono essere utilizzate le istruzioni della Z80 che pongono il contenuto del registro B sulla parte alta (A15-A8) del bus degli indirizzi (ad esempio OUT (C),n).

Tastiera

Per ottenere il reset completo del computer è necessario premere in sequenza i tasti [CTRL][SHIFT][ESC]. Tutti i tasti, ad eccezione di quelli del pad numerico, sono dotati di autorepeat controllato dal firmware.

La pressione del tasto [ESC] sospende momentaneamente l'esecuzione del programma, che può essere ripresa con la pressione di un qualsiasi tasto che non sia [ESC]. Premendo invece nuovamente [ESC] si interrompe definitivamente l'esecuzione.

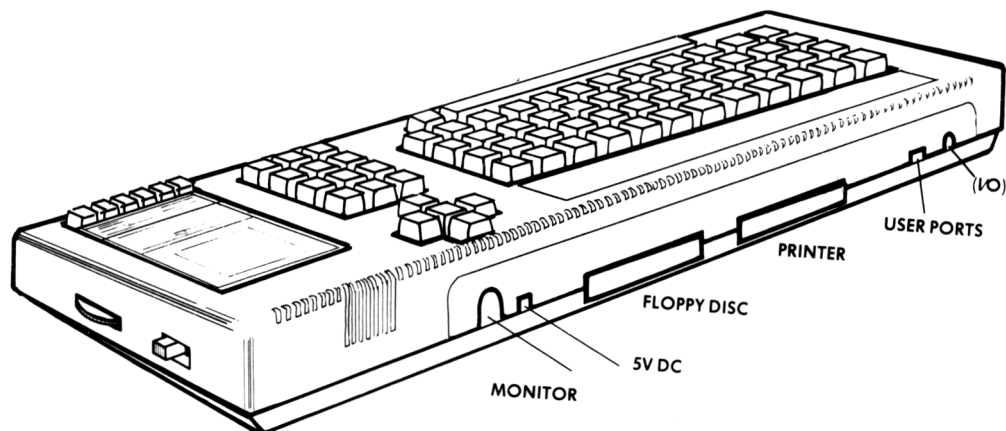
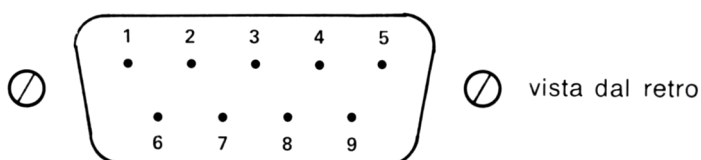
Il tasto [CAPS LOCK] ha la funzione equivalente a quella del tasto 'Fissa maiuscole' delle macchine da scrivere. È sufficiente premerlo una sola volta perchè tutte le lettere battute risultino scritte in maiuscolo. Questa funzione non ha effetto sui tasti numerici in quanto i simboli riportati sulla parte superiore di detti tasti non verranno rappresentati. Per ottenere le lettere maiuscole e i simboli della fila superiore è sufficiente premere i tasti [SHIFT] e [CAPS LOCK] una sola volta.

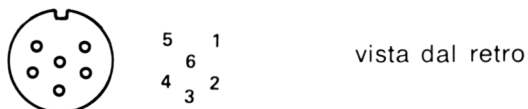
Il cursore di COPY viene separato da quello normale con la pressione contemporanea del tasto [SHIFT] e di uno di quelli di spostamento del cursore. Tramite la pressione del tasto [COPY] è possibile inserire nel buffer di input i caratteri che si trovano alla destra del cursore di COPY.

I tasti di movimento del cursore possono essere utilizzati per determinare il punto dello schermo da cui deve partire l'input da tastiera. Tale posizione viene fissata solo con la pressione del tasto [ENTER]. I caratteri inseriti andranno a sovrapporsi a quelli eventualmente già presenti sullo schermo alla posizione scelta.

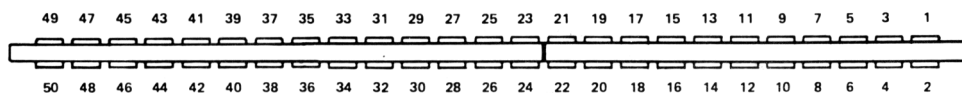
Premendo il tasto [DEL] si cancella il carattere che si trova alla sinistra del cursore, con il tasto [CLR] quello che si trova alla destra.

APPENDICE V

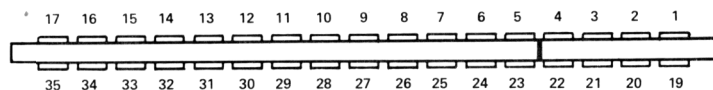
CONNESSIONI POSTERIORI

Porta joystick (DIN-9 PIN)

Pin 1	su	Pin 6	fuoco 2
Pin 2	giù	Pin 7	fuoco 1
Pin 3	sinistra	Pin 8	common
Pin 4	destra	Pin 9	com 2
Pin 5	non connesso		

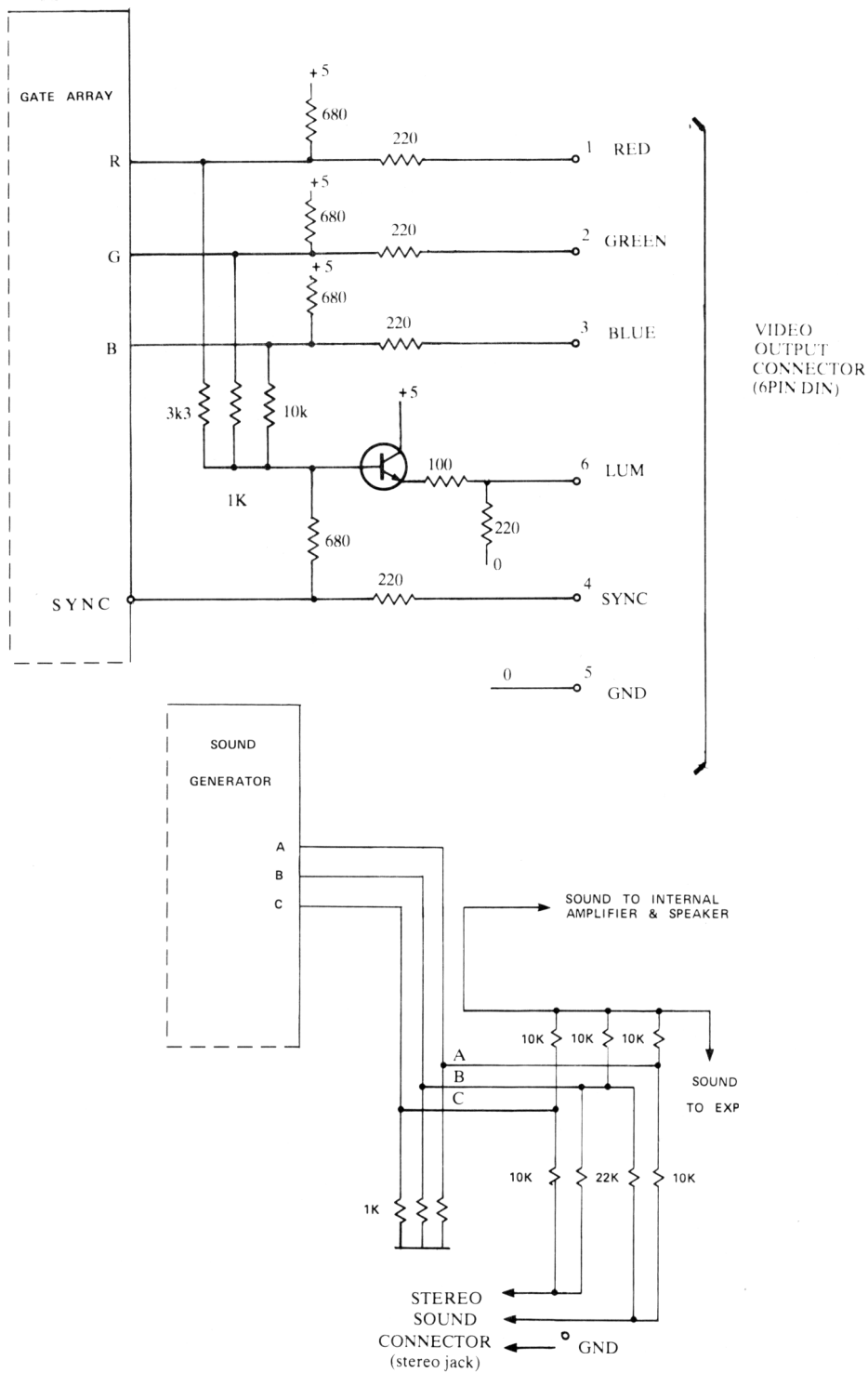
Connettore video (DIN-6 PIN)

Pin 1	rosso	Pin 4	sync
Pin 2	verde	Pin 5	massa
Pin 3	blu	Pin 6	lum

PORTA DI ESPANSIONE — CONNETTORE A 50 VIE

vista dal retro

PIN	Segnale	PIN	Segnale	PIN	Segnale
PIN 1	SOUND	PIN 18	A0	PIN 35	$\overline{\text{INT}}$
PIN 2	GND	PIN 19	D7	PIN 36	$\overline{\text{NMI}}$
PIN 3	A15	PIN 20	D6	PIN 37	$\overline{\text{BUSRD}}$
PIN 4	A14	PIN 21	D5	PIN 38	$\overline{\text{BUSAK}}$
PIN 5	A13	PIN 22	D4	PIN 39	READY
PIN 6	A12	PIN 23	D3	PIN 40	$\overline{\text{BUS RESET}}$
PIN 7	A11	PIN 24	D2	PIN 41	$\overline{\text{RESET}}$
PIN 8	A10	PIN 25	D1	PIN 42	$\overline{\text{ROMEN}}$
PIN 9	A9	PIN 26	D0	PIN 43	ROMDIS
PIN 10	A8	PIN 27	+ 5v	PIN 44	$\overline{\text{RAMRD}}$
PIN 11	A7	PIN 28	$\overline{\text{MREQ}}$	PIN 45	RAMDIS
PIN 12	A6	PIN 29	$\overline{\text{M1}}$	PIN 46	CURSOR
PIN 13	A5	PIN 30	$\overline{\text{RFSH}}$	PIN 47	L. PEN
PIN 14	A4	PIN 31	$\overline{\text{IORQ}}$	PIN 48	$\overline{\text{EXP}}$
PIN 15	A3	PIN 32	$\overline{\text{RD}}$	PIN 49	GND
PIN 16	A2	PIN 33	$\overline{\text{WR}}$	PIN 50	ϕ
PIN 17	A1	PIN 34	$\overline{\text{HALT}}$		

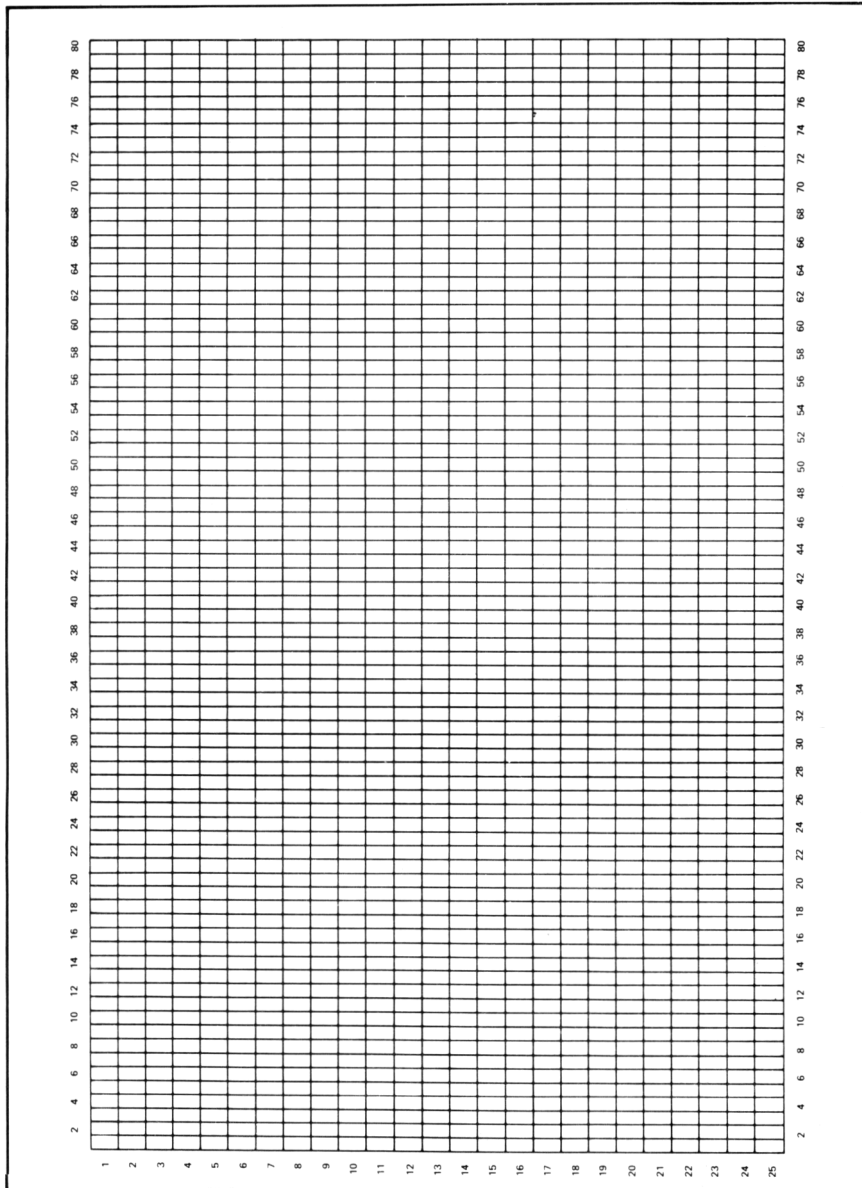
PORTA STAMPANTE — CONNETTORE A 34 VIE

vista dal retro

PIN	Segnale	PIN	Segnale
PIN 1	$\overline{\text{STROBE}}$	PIN 19	GND
PIN 2	D0	PIN 20	GND
PIN 3	D1	PIN 21	GND
PIN 4	D2	PIN 22	GND
PIN 5	D3	PIN 23	GND
PIN 6	D4	PIN 24	GND
PIN 7	D5	PIN 25	GND
PIN 8	D6	PIN 26	GND
PIN 9	D7	PIN 27	GND
PIN 11	BUSY	PIN 28	GND
PIN 14	GND	PIN 33	GND
PIN 16	GND	All other pins NC	

GATE ARRAY
R
G
B
SYNC
+5
680
220
1 RED
2 GREEN
3 BLUE
3k3
10k
1K
100
6 LUM
220
0
4 SYNC
0
5 GND
VIDEO
OUTPUT
CONNECTOR
(6PIN DIN)
SOUND
GENERATOR
A
B
C
SOUND TO INTERNAL
AMPLIFIER & SPEAKER
10K
22K
1K
SOUND
TO EXP
STEREO
SOUND
CONNECTOR
(stereo jack)
GND

APPENDICE VI

Griglia per il testo e le finestre
Modo 0 / 20 colonne

Griglia per il testo e le finestre
Modo 1 / 40 colonne

Griglia per il testo e le finestre
Modo 2 / 80 colonne

APPENDICE VII

NOTE E PERIODI DI TONO

Nella tabella che segue sono indicati i Periodi di tono consigliati per le note della normale scala musicale su tutte le 8 ottave.

La frequenza prodotta non corrisponde esattamente alla frequenza richiesta in quanto il periodo deve essere definito da un intero. L'errore relativo corrisponde al rapporto:

(frequenza richiesta - frequenza effettiva)/frequenza richiesta

Nota	Frequenza	Periodo	Errore relativo	
C	32.703	3822	-0.007%	
C#	34.648	3608	+0.007%	
D	36.708	3405	-0.007%	
D#	38.891	3214	-0.004%	
E	41.203	3034	+0.009%	
F	43.654	2863	-0.016%	**Ottava –3**
F#	46.249	2703	+0.009%	
G	48.999	2551	-0.002%	
G#	51.913	2408	+0.005%	
A	55.000	2273	+0.012%	
A#	58.270	2145	-0.008%	
B	61.735	2025	+0.011%	

Nota	Frequenza	Periodo	Errore relativo	
C	65.406	1911	-0.007%	
C#	69.296	1804	+0.007%	
D	73.416	1703	+0.022%	
D#	77.782	1607	-0.004%	
E	82.407	1517	+0.009%	
F	87.307	1432	+0.019%	**Ottava –2**
F#	92.499	1351	-0.028%	
G	97.999	1276	+0.037%	
G#	103.826	1204	+0.005%	
A	110.000	1136	-0.032%	
A#	116.541	1073	+0.039%	
B	123.471	1012	-0.038%	

Nota	Frequenza	Periodo	Errore relativo	
C	130.813	956	+0.046%	
C#	138.591	902	+0.007%	
D	146.832	851	-0.037%	
D#	155.564	804	+0.058%	
E	164.814	758	-0.057%	
F	174.614	716	+0.019%	
F#	184.997	676	+0.046%	**Ottava –1**
G	195.998	638	+0.037%	
G#	207.652	602	+0.005%	
A	220.000	568	-0.032%	
A#	233.082	536	-0.055%	
B	246.942	506	-0.038%	

Nota	Frequenza	Periodo	Errore relativo	
C	261.626	478	+0.046%	Middle C
C#	277.183	451	+0.007%	
D	293.665	426	+0.081%	
D#	311.127	402	+0.058%	
E	329.628	379	-0.057%	
F	349.228	358	+0.019%	**Ottava 0**
F#	369.994	338	+0.046%	
G	391.995	319	+0.037%	
G#	415.305	301	+0.005%	
A	440.000	284	-0.032%	International A
A#	466.164	268	-0.055%	
B	493.883	253	-0.038%	

Nota	Frequenza	Periodo	Errore relativo	
C	523.251	239	+0.046%	
C#	554.365	225	-0.215%	
D	587.330	213	+0.081%	
D#	622.254	201	+0.058%	
E	659.255	190	+0.206%	
F	698.457	179	+0.019%	**Ottava 1**
F#	739.989	169	+0.046%	
G	783.991	159	-0.277%	
G#	830.609	150	-0.328%	
A	880.000	142	-0.032%	
A#	932.328	134	-0.055%	
B	987.767	127	+0.356%	

Nota	Frequenza	Periodo	Errore relativo	
C	1046.502	119	-0.374%	
C#	1108.731	113	+0.229%	
D	1174.659	106	-0.390%	
D#	1244.508	100	-0.441%	
E	1318.510	95	+0.206%	
F	1396.913	89	-0.543%	Ottava 2
F#	1479.978	84	-0.548%	
G	1567.982	80	+0.350%	
G#	1661.219	75	-0.328%	
A	1760.000	71	-0.032%	
A#	1864.655	67	-0.055%	
B	1975.533	63	-0.435%	
Nota	**Frequenza**	**Periodo**	**Errore relativo**	
C	2093.004	60	+0.462%	
C#	2217.461	56	-0.662%	
D	2349.318	53	-0.390%	
D#	2489.016	50	-0.441%	
E	2637.021	47	-0.855%	
F	2793.826	45	+0.574%	Ottava 3
F#	2959.955	42	-0.548%	
G	3135.963	40	+0.350%	
G#	3322.438	38	+0.992%	
A	3520.000	36	+1.357%	
A#	3729.310	34	+1.417%	
B	3951.066	32	+1.134%	
Nota	**Frequenza**	**Periodo**	**Errore relativo**	
C	4186.009	30	+0.462%	
C#	4434.922	28	-0.662%	
D	4698.636	27	+1.469%	
D#	4978.032	25	-0.441%	
E	5274.041	24	+1.246%	
F	5587.652	22	-1.685%	Ottava 4
F#	5919.911	21	-0.548%	
G	6271.927	20	+0.350%	
G#	6644.875	19	+0.992%	
A	7040.000	18	+1.357%	
A#	7458.621	17	+1.417%	
B	7902.133	16	+1.134%	

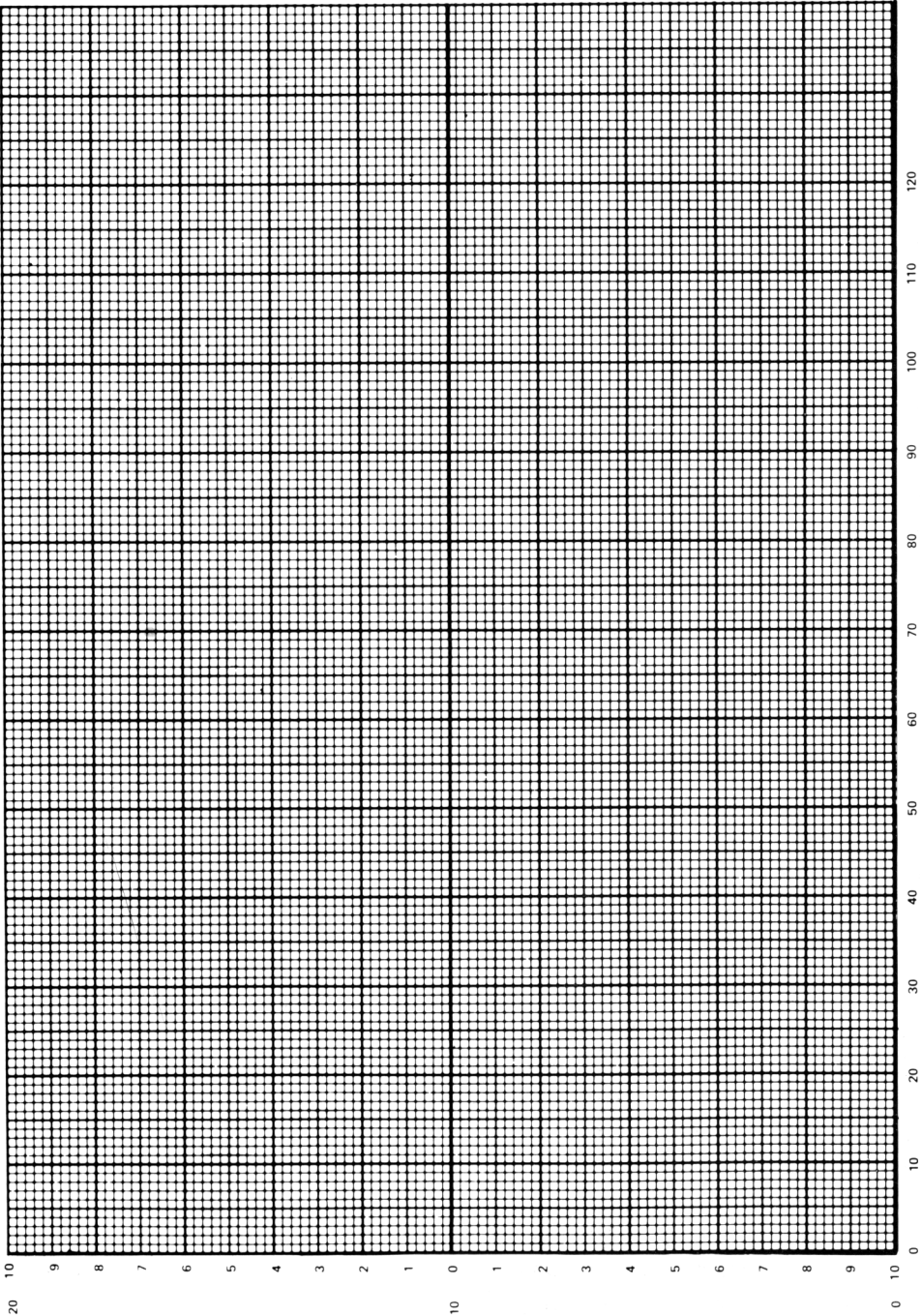
Do centrale

Tutti i valori sono calcolati secondo le formule:

frequenza = 440 * (2↑(ottava +(10−n)/12))

Periodo = ROUND (125000/frequenza)

dove n = 1 per C; 2 per C#; 3 per D; e cosi via.

Tracciato per il disegno degli inviluppi

APPENDICE VIII

CODICI DI ERRORE E PAROLE RISERVATE

Codici e messaggi di errore

Quando il BASIC incontra uno statement che non è in grado di comprendere e processare, interrompe l'esecuzione e visualizza un messagio d'errore, che indica genericamente il tipo di errore che si è verificato.

Gli errori più frequentemente commessi sono quelli di battitura, identificati dal messaggio 'Syntax error'. Questi durante l'esecuzione del programma provocano la visualizzazione della riga in cui l'errore si trova, pronta per essere corretta. In modo diretto, si assume che la riga in cui si trova l'errore sia ancora visibile sullo schermo dopo la visualizzazione del 'Syntax error'.

Nel caso in cui un comando ON ERROR GOTO sia inserito all'inizio del programma, al verificarsi di un errore l'esecuzione proseguirà dalla riga specificata. Nell'esempio seguente, si indica al computer di saltare alla riga 1000 in caso di errore:

```
10 ON ERROR GOTO 1000
programma
1000 PRINT CHR$(7) : MODE 2: INK 1,0: INK 0,9: CLS: LIST
```

In questo caso il CPC 464 genererà un beep, pulirà lo schermo, selezionerà una coppia di colori per ottenere la rappresentazione ottimale delle 80 colonne e listerà il programma. Se l'errore è di sintassi, la riga contenente l'errore verrà posta al termine del listato del programma e il CPC 464 entrerà in modo edit per permetterne la correzione.

Ricordate di inserire un comando END tra la fine del programma e la routine di gestione dell'errore che inizia alla riga 1000.

Tutti i messaggi di errore generati dal BASIC sono listati qui di seguito, in ordine di codice. Ogni messaggio è seguito da una breve spiegazione delle cause che lo possono aver generato:

1 — Unexpected NEXT

È stato incontrato un NEXT che non è preceduto da alcun comando FOR, oppure la variabile di controllo indicata nel comando NEXT non trova corrispondenza con quella dichiarata nel FOR.

2 — Syntax error

Il BASIC non è in grado di interpretare l'istruzione incontrata.

3 — Unexpected RETURN

È stato trovato un comando RETURN al di fuori di una subroutine.

4 — DATA exhausted

Un comando READ ha cercato di leggere dati oltre la fine dell'ultimo DATA.

5 — Improper argument

È un errore di tipo generico. Il valore dell'argomento di una funzione o il parametro di un comando è in qualche modo non valido.

6 — Overflow

Il risultato di un'operazione matematica ha superato la capacità di rappresentazione del CPC 464. L'overflow può verificarsi sia con numeri in virgola mobile, nel caso che qualche operazione fornisca un risultato maggiore di 1.7E-38, oppure durante un tentativo di convertire un numero da notazione in virgola mobile in notazione intera su 16 bit.

7 — Memory full

Il programma corrente è troppo grande per essere contenuto in memoria, oppure le strutture di controllo (GOSUB, WHILE e FOR) sono eccessivamente nidificate.

Il comando MEMORY provoca un errore di questo tipo nel caso in cui l'utente cerchi di abbassare eccessivamente il limite della memoria disponibile al BASIC. Notate che l'apetura di un file su cassetta corrisponde all'allocazione di una zona di memoria quale buffer, e questo riduce la quantità di memoria disponibile.

8 — Linee does not exist

La riga di programma specificata non è stata trovata.

9 — Subscrit out of range

L'indice di un vettore è troppo piccolo o troppo grande.

10 — Array already dimensioned

Uno degli array indicati in uno statement DIM è già stato dimensionato.

11 — Division by zero

È stata eseguita una divisione per 0. Può occorrere in una divisione fra numeri reali, interi o in un elevamento a potenza.

12 — Invalid direct comand

L'ultimo comando inserito in modo diretto non può essere eseguito.

13 — Type mismatch

Un dato numerico è stato inserito dove ne era richiesto uno alfanumerico, o viceversa.

14 — String space full

La zona di memoria riservata per la memorizzazione delle stringhe è stata totalmente occupata.

15 — String too long

La stringa supera la lunghezza massima di 255 caratteri. Può occorrere durante il concatenamento di più stringhe.

16 — String expression too complex

Le operazioni sulle stringhe possono generare vari risultati intermedi. Quando il numero di questi risultati parziali è troppo grande, il BASIC non è in grado di risolvere l'espressione e genera l'errore.

17 — Cannot CONTinue

Per una qualche ragione il BASIC non è in grado di proseguire l'esecuzione dopo che questa è stata interrotta con uno STOP, con [ESC][ESC] o a causa di un errore.

18 — Unknown user function

La funzione richiamata con un FN non è stata precedentemente definita con una DEF FN.

19 — RESUME missing

L'esecuzione del programma è terminata pur essendo all'interno di una routine di gestione dell'errore.

20 — Unexpected RESUME

Il comando RESUME può essere inserito solo in una routine richiamata da un ON ERROR GOTO.

21 — Direct command found

Durante il caricamento da cassetta è stata incontrata una riga priva del numero di riga.

22 — Operand missing

Il BASIC ha incontrato una espressione incompleta.

23 — Line too long

Dopo la conversione nel formato interno del BASIC la lunghezza della riga eccede i limiti consentiti.

24 — EOF met

Il programma ha cercato di leggere un file oltre la sua fine.

25 — File type error

Il file che è stato aperto non è del tipo corretto. Il comando OPENIN permette di leggere file testo in formato ASCII. I comandi LOAD, RUN ecc. sono adatti per leggere i file prodotti dalla SAVE.

26 — NEXT missing

Un ciclo FOR/NEXT non è stato chiuso con il NEXT.

27 — File already open

Un comando OPENIN o OPENOUT è stato eseguito specificandto un file già aperto.

28 — Unknown command

Il BASIC non trova riscontro alla chiamata di un comando esterno.

29 — WEND missing

Un ciclo WHILE/WEND non è stato chiuso con lo WEND.

30 — Unexpected WEND

Il comando WEND incontrato non ha riscontro con alcun comando WHILE precedente.

Parole riservate del Basic

Segue la lista delle parole chiave del BASIC, che non possono essere utilizzate come nomi di variabili:

ABS,AFTER,AND,ASC,ATN,AUTO

BIN$,BORDER

CALL,CAT,CHAIN,CHR$,CINT,CLEAR,CLG,CLOSEIN,CLOSEOUT,CLS,CONT, COS,CREAL

DATA,DEF,DEFINT,DEFREAL,DEFSTR,DEG,DELETE,DI,DIM,DRAW,DRAWR, EDIT,EI,ELSE,END,ENT,ENV,EOF,ERASE,ERL,ERR,ERROR,EVERY, EXP,FIX,FN,FOR,FRE

FIX,FN,FOR,FRE

GOSUB,GOTO

HEX$,HIMEN

IF,INK,INKEY,INKEY$,INP,INPUT,INSTR,INT

JOY

KEY

LEFT$,LEN,LET,LINE,LIST,LOAD,LOCATE,LOG,LOG10,LOWER$

MAX,MEMORY,MERGE,MID$,MIN,MOD,MODE,MOVE,MOVER

NEXT,NEW,NOT

ON,ON BREAK,ON ERROR GOTO,ON SQ,OPENIN,OPENOUT,OR,ORIGIN,OUT

PAPER,PEEK,PEN,PI,PLOT,PLOTR,POKE,POS,PRINT

RAD,RANDOMIZE,READ,RELEASE,REM,REMAIN,RENUM,RESTORE,
RESUM,RETURN,RIGHT$,RND,ROUND,RUN

SAVE,SGN,SIN,SOUND,SPACE$,SPC,SPEED,SQ,STEP,STOP,STR$,
STRING$,SWAP,SYMBOL

TAB,TAG,TAGOFF,TAN,TEST,TESTR,THEN,TIME,TO,TROFF,TRON

UNT,UPPER$,USING

VAL,VPOS

WAIT,WEND,WHILE,WIDTH,WINDOW,WRITE

XOR,XPOS

YPOS

ZONE

INDICE ANALITICO